CHAPTER 6

Speech Signal Representations

This chapter presents several representations for speech signals useful in speech coding, synthesis and recognition. The central theme is the decomposition of the speech signal as a source passed through a linear time-varying filter. This filter can be derived from models of speech production based on the theory of acoustics where the source represents the air flow at the vocal cords, and the filter represents the resonances of the vocal tract which change over time. Such a source-filter model is illustrated in Figure 6.1. We describe methods to compute both the source or *excitation* $e[n]$ and the filter $h[n]$ from the speech signal $x[n]$.

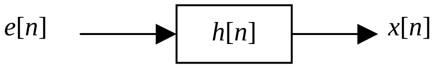

Figure 6.1 Basic source-filter model for speech signals.

To estimate the filter we present methods inspired by speech production models (such as linear predictive coding and cepstral analysis) as well as speech perception models (such as mel-frequency cepstrum). Once the filter has been estimated, the source can be obtained by passing the speech signal through the inverse filter. Separation between source and filter is one of the most difficult challenges in speech processing.

It turns out that phoneme classification (either by human or by machines) is mostly dependent on the characteristics of the filter. Traditionally, speech recognizers estimate the filter characteristics and ignore the source. Many speech synthesis techniques use a source-filter model because it allows flexibility in altering the pitch and the filter. Many speech coders also use this model because it allows a low bit rate.

We first introduce the spectrogram as a representation of the speech signal that highlights several of its properties and describe the short-time Fourier analysis, which is the basic tool to build the spectrograms of Chapter 2. We then introduce several techniques used to separate source and filter: LPC and cepstral analysis, perceptually motivated models, formant tracking, and pitch tracking.

6.1. SHORT-TIME FOURIER ANALYSIS

In Chapter 2, we demonstrated how useful *spectrograms* are to analyze phonemes and their transitions. A spectrogram of a time signal is a special two-dimensional representation that displays time in its horizontal axis and frequency in its vertical axis. A gray scale is typically used to indicate the energy at each point (*t*, *f*) with white representing low energy and black high energy. In this section we cover short-time Fourier analysis, the basic tool with which to compute them.

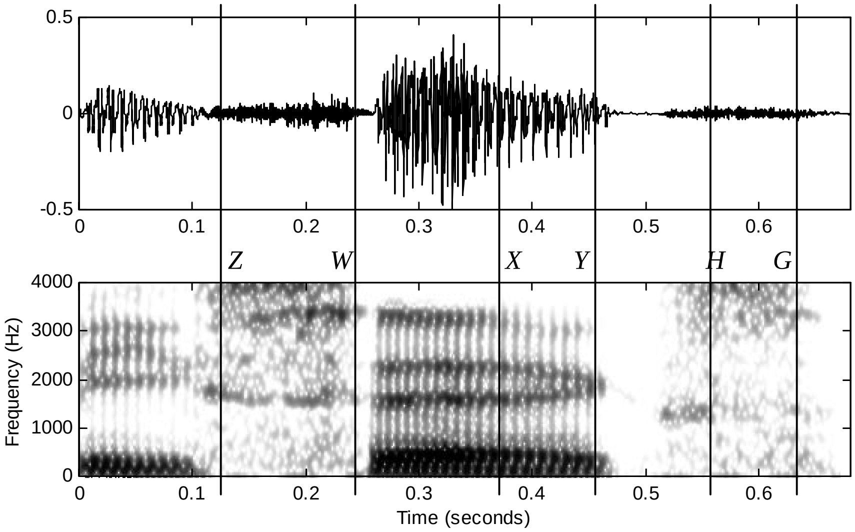

Figure 6.2 (a) Waveform with (b) its corresponding wideband spectrogram. Darker areas mean higher energy for that time and frequency. Note the vertical lines spaced by pitch peri-

ods.

The idea behind a spectrogram, such as that in Figure 6.2, is to compute a Fourier transform every 5 milliseconds or so, displaying the energy at each time/frequency point. Since some regions of speech signals shorter than, say, 100 milliseconds often appear to be periodic, we use the techniques discussed in Chapter 5. However, the signal is no longer periodic when longer segments are analyzed, and therefore the exact definition of Fourier transform cannot be used. Moreover, that definition requires knowledge of the signal for infinite time. For both reasons, a new set of techniques called *short-time analysis*, are proposed. These techniques decompose the speech signal into a series of short segments, referred to as *analysis frames*, and analyze each one independently.

In Figure 6.2 (a), note the assumption that the signal can be approximated as periodic within *X* and *Y* is reasonable. In regions (*Z*, *W*) and (*H*, *G*), the signal is not periodic and looks like *random noise*. The signal in (*Z*, *W*) appears to have different noisy characteristics than those of segment (*H*, *G*). The use of an analysis frame implies that the region is short enough for the behavior (periodicity or noise-like appearance) of the signal to be approximately constant. If the region where speech seems periodic is too long, the pitch period is not constant and not all the periods in the region are similar. In essence, the speech region has to be short enough so that the signal is *stationary* in that region: *i.e.*, the signal characteristics (whether periodicity or noise-like appearance) are uniform in that region. A more formal definition of stationarity is given in Chapter 5.

Similarly to the filterbanks described in Chapter 5, given a speech signal $x[n]$, we define the short-time signal $x_m[n]$ of frame *m* as

$$x_m[n] = x[n]w_m[n] \tag{6.1}$$

the product of $x[n]$ by a *window* function $w_m[n]$, which is zero everywhere except in a small region.

While the window function can have different *values* for different frames *m*, a popular choice is to keep it constant for all frames:

$$w_m[n] = w[m-n] \tag{6.2}$$

where $w[n] = 0$ for $|n| > N/2$. In practice, the window length is on the order of 20 to 30 ms.

With the above framework, the short-time Fourier representation for frame *m* is defined as

$$X_m(e^{j\omega}) = \sum_{n=-\infty}^{\infty} x_m[n]e^{-j\omega n} = \sum_{n=-\infty}^{\infty} w[m-n]x[n]e^{-j\omega n} \tag{6.3}$$

with all the properties of Fourier transforms studied in Chapter 5.

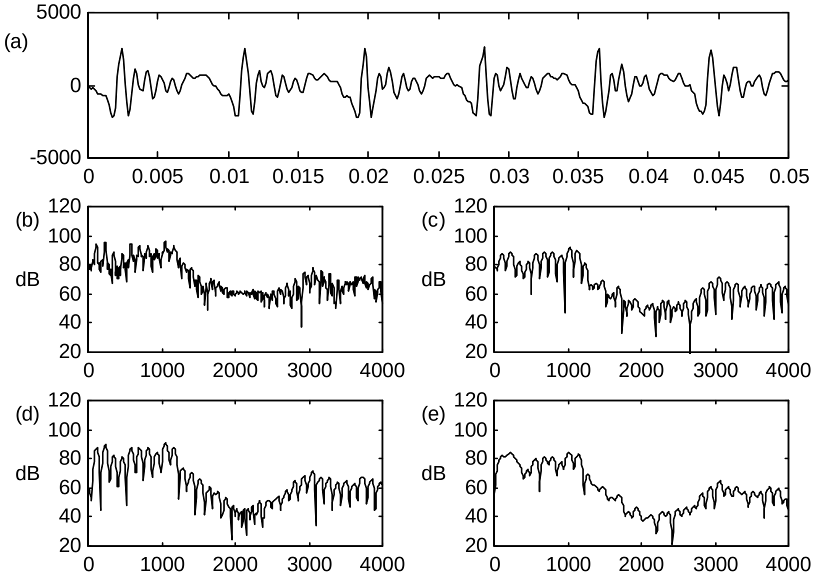

Figure 6.3 Short-time spectrum of male voiced speech (vowel /ah/ with local pitch of 110Hz): (a) time signal, spectra obtained with (b) 30ms rectangular window and (c) 15 ms rectangular window, (d) 30 ms Hamming window, (e) 15ms Hamming window. The window lobes are not visible in (e), since the window is shorter than 2 times the pitch period. Note the spectral leakage present in (b).

In Figure 6.3 we show the short-time spectrum of voiced speech. Note that there are a number of peaks in the spectrum. To interpret this, assume the properties of $x_m[n]$ persist outside the window, and that, therefore, the signal is periodic with period M in the true sense. In this case, we know (see Chapter 5) that its spectrum is a sum of impulses

$$X_m(e^{j\omega}) = \sum_{k=-\infty}^{\infty} X_m[k]\delta(\omega - 2\pi k / M) \tag{6.4}$$

Given that the Fourier transform of $w[n]$ is

$$W(e^{j\omega}) = \sum_{n=-\infty}^{\infty} w[n]e^{-j\omega n} \tag{6.5}$$

so that the transform of $w[m-n]$ is $W(e^{-j\omega})e^{-j\omega m}$. Therefore, using the convolution property, the transform of $x[n]w[m-n]$ for fixed m is the convolution in the frequency domain

$$X_m(e^{j\omega}) = \sum_{k=-\infty}^{\infty} X_m[k] W(e^{j(\omega-2\pi k/N)}) e^{j(\omega-2\pi k/N)m} \tag{6.6}$$

which is a sum of weighted $W(e^{j\omega})$, shifted on every harmonic, the narrow peaks seen in Figure 6.3 (b) with a rectangular window. The short-time spectrum of a periodic signal exhibits peaks (equally spaced $2\pi / M$ apart) representing the harmonics of the signal. We estimate $X_m[k]$ from the short-time spectrum $X_m(e^{j\omega})$, and we see the importance of the length and choice of window.

Equation (6.6) indicates that one cannot recover $X_m[k]$ by simply retrieving $X_m(e^{j\omega})$, although the approximation can be reasonable if there is a small value of λ such that

$$W(e^{j\omega}) \approx 0 \text{ for } \left|\omega - \omega_k\right| > \lambda \tag{6.7}$$

which is the case outside the main lobe of the window's frequency response.

Recall from Section 5.4.2.1 that, for a rectangular window of length *N*, $\lambda = 2\pi / N$. Therefore, Eq. (6.7) is satisfied if $N \geq M$, *i.e.*, the rectangular window contains at least one pitch period. The width of the main lobe of the window's frequency response is inversely proportional to the length of the window. The pitch period in Figure 6.3 is *M* = 71 at a sampling rate of 8 kHz. A shorter window is used in Figure 6.3 (c), which results in wider analysis lobes, though still visible.

Also recall from Section 5.4.2.2 that for a Hamming window of length *N*, $\lambda = 4\pi / N$: twice as wide as that of the rectangular window, which entails $N \geq 2M$. Thus, for Eq. (6.7) to be met, a Hamming window must contain at least two pitch periods. The lobes are visible in Figure 6.3 (d) since *N* = 240, but they are not visible in Figure 6.3 (e) since *N* = 120, and $N < 2M$.

In practice, one cannot know what the pitch period is ahead of time, which often means you need to prepare for the lowest pitch period. A low-pitched voice with a $F_0 = 50\,\text{Hz}$ requires a rectangular window of at least 20 ms and a Hamming window of at least 40 ms for the condition in Eq. (6.7) to be met. If speech is non-stationary within 40ms, taking such a long window implies obtaining an average spectrum during that segment instead of several distinct spectra. For this reason, the rectangular window provides better *time resolution* than the Hamming window. Figure 6.4 shows analysis of female speech for which shorter windows are feasible.

But the frequency response of the window is not completely zero outside its main lobe, so one needs to see the effects of this incorrect assumption. From Section 5.4.2.1 note that the second lobe of a rectangular window is only approximately 17 dB below the main lobe. Therefore, for the k^{th} harmonic the value of $X_m(e^{j2\pi k/M})$ contains not $X_m[k]$, but also a weighted sum of $X_m[l]$. This phenomenon is called *spectral leakage* because the amplitude of one harmonic leaks over the rest and masks its value. If the signal's spectrum is white, spectral leakage does not cause a major problem, since the effect of the second lobe on a harmonic is only $10\log_{10}(1+10^{-17/10}) = 0.08\text{dB}$. On the other hand, if the signal's spectrum decays more quickly in frequency than the decay of the window, the spectral leakage results in inaccurate estimates.

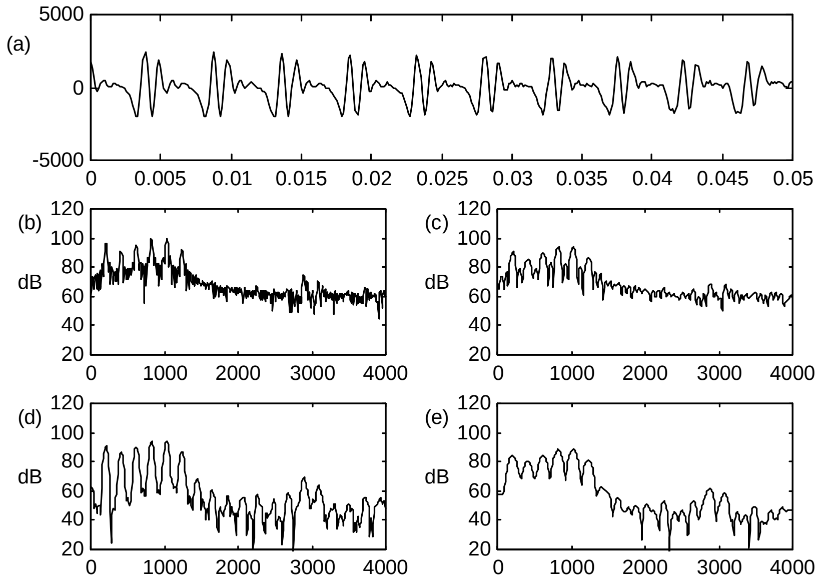

Figure 6.4 Short-time spectrum of female voiced speech (vowel /aa/ with local pitch of 200Hz): (a) time signal, spectra obtained with (b) 30 ms rectangular window and (c) 15 ms rectangular window, (d) 30 ms Hamming window, (e) 15 ms Hamming window. In all cases the window lobes are visible, since the window is longer than 2 times the pitch period. Note the spectral leakage present in (b) and (c).

From Section 5.4.2.2, observe that the second lobe of a Hamming window is approximately 43 dB, which means that the spectral leakage effect is much less pronounced. Other windows, such as Hanning, or triangular windows, also offer less spectral leakage than the rectangular window. This important fact is the reason why, despite their better time resolution, rectangular windows are rarely used for speech analysis. In practice, window lengths are on the order of 20 to 30 ms. This choice is a compromise between the stationarity assumption and the frequency resolution.

In practice, the Fourier transform in Eq. (6.3) is obtained through an FFT. If the window has length *N*, the FFT has to have a length greater than or equal to *N*. Since FFT algorithms often have lengths that are powers of 2 ($L = 2^R$), the windowed signal with length *N* is augmented with $(L-N)$ zeros either before, after, or both. This process is called *zero-padding*. A larger value of *L* provides a finer description of the discrete Fourier transform; but it does not increase the analysis frequency resolution: this is the sole mission of the window length *N*.

In Figure 6.3, observe the broad peaks, resonances or formants, which represent the filter characteristics. For voiced sounds there is typically more energy at low frequencies than at high frequencies, also called *roll-off*. It is impossible to determine exactly the filter characteristics, because we know only samples at the harmonics, and we have no knowledge of the values in between. In fact, the resonances are less obvious in Figure 6.4 because the harmonics sample the spectral envelope less densely. For high-pitched female speakers and children, it is even more difficult to locate the formant resonances from the short-time spectrum.

Figure 6.5 shows the short-time analysis of unvoiced speech, for which no regularity is observed.

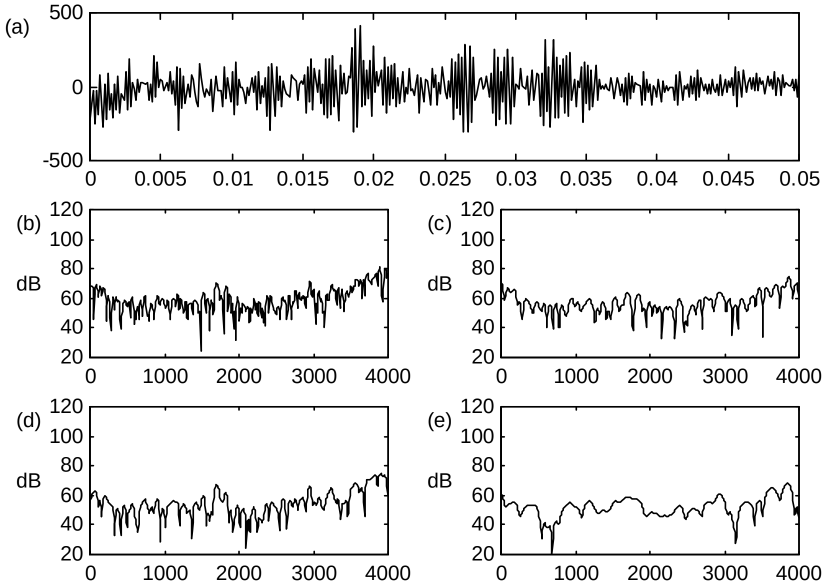

Figure 6.5 Short-time spectrum of unvoiced speech. (a) time signal, (b) 30 ms rectangular window (c) 15 ms rectangular window, (d) 30 ms Hamming window (e) 15 ms Hamming window.

6.1.1. Spectrograms

Since the spectrogram displays just the energy and not the phase of the short-term Fourier transform, we compute the energy as

$$\log |X[k]|^2 = \log\left(X_r^2[k] + X_i^2[k]\right) \tag{6.8}$$

with this value converted to a gray scale according to Figure 6.6. Pixels whose values have not been computed are interpolated. The slope controls the contrast of the spectrogram, while the saturation points for white and black control the dynamic range.

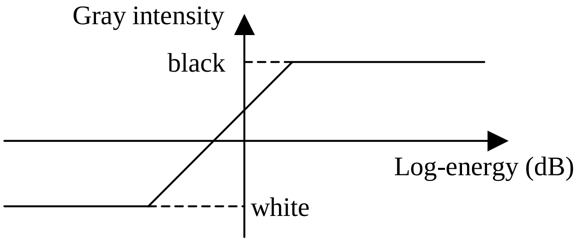

Figure 6.6 Conversion between log-energy values (in the *x*-axis) and gray scale (in the *y*-axis). Larger log-energies correspond to a darker gray color. There is a linear region for which more log-energy corresponds to darker gray, but there is saturation at both ends. Typically there is 40 to 60 dB between the pure white and the pure black.

There are two main types of spectrograms: *narrow-band* and *wide-band*. Wide-band spectrograms use relatively short windows (< 10 ms) and thus have good time resolution at the expense of lower frequency resolution, since the corresponding filters have wide bandwidths (> 200 Hz) and the harmonics cannot be seen. Note the vertical stripes in Figure 6.2, due to the fact that some windows are centered at the high part of a pitch pulse, and others in between have lower energy. Spectrograms can aid in determining formant frequencies and fundamental frequency, as well as voiced and unvoiced regions.

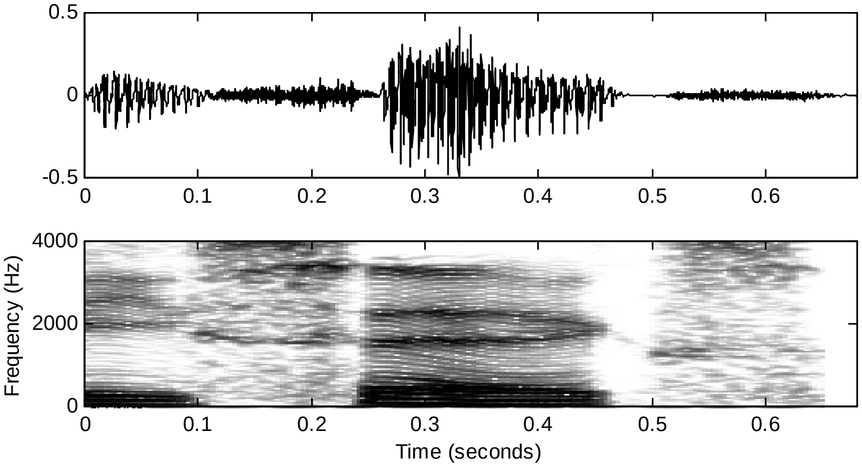

Figure 6.7 Waveform (a) with its corresponding narrowband spectrogram (b). Darker areas mean higher energy for that time and frequency. The harmonics can be seen as horizontal lines spaced by fundamental frequency. The corresponding wideband spectrogram can be seen in Figure 6.2.

Narrow-band spectrograms use relatively long windows (> 20 ms), which lead to filters with narrow bandwidth (< 100 Hz). On the other hand, time resolution is lower than for wide-band spectrograms (see Figure 6.7). Note that the harmonics can be clearly seen, because some of the filters capture the energy of the signal's harmonics, and filters in between have little energy.

Some implementation details also need to be taken into account. Since speech signals are real, the Fourier transform is Hermitian, and its power spectrum is also even. Thus, it is only necessary to display values for $0 \le k \le N/2$ for N even. In addition, while the traditional spectrogram uses a gray scale, a color scale can also be used, or even a 3-D representation. In addition, to make the spectrograms easier to read, sometimes the signal is first preemphasized (typically with a first-order difference FIR filter) to boost the high frequencies to counter the roll-off of natural speech.

By inspecting both narrow-band and wide-band spectrograms, we can learn the filter's magnitude response and whether the source is voiced or not. Nonetheless it is very difficult to separate source and filter due to nonstationarity of the speech signal, spectral leakage, and the fact that only the filter's magnitude response can be known at the signal's harmonics.

6.1.2. Pitch-Synchronous Analysis

In the previous discussion, we assumed that the window length is fixed, and we saw the tradeoffs between a window that contained several pitch periods (narrow-band spectrograms) and a window that contained less than a pitch period (wide-band spectrograms). One possibility is to use a rectangular window whose length is exactly one pitch period; this is called *pitch-synchronous* analysis. To reduce spectral leakage a tapering window, such as Hamming or Hanning, can be used, with the window covering exactly two pitch periods. This latter option provides a very good compromise between time and frequency resolution. In this representation, no stripes can be seen in either time or frequency. The difficulty in computing pitch synchronous analysis is that, of course, we need to know the local pitch period, which, as we see in Section 6.7, is not an easy task.

6.2. ACOUSTICAL MODEL OF SPEECH PRODUCTION

Speech is a sound wave created by vibration that is propagated in the air. Acoustic theory analyzes the laws of physics that govern the propagation of sound in the vocal tract. Such a theory should consider three-dimensional wave propagation, the variation of the vocal tract shape with time, losses due to heat conduction and viscous friction at the vocal tract walls, softness of the tract walls, radiation of sound at the lips, nasal coupling and excitation of sound. While a detailed model that considers all of the above is not yet available, some models provide a good approximation in practice, as well as a good understanding of the physics involved.

6.2.1. Glottal Excitation

As discussed in Chapter 2, the vocal cords constrict the path from the lungs to the vocal tract. This is illustrated in Figure 6.8. As lung pressure is increased, air flows out of the lungs and through the opening between the vocal cords (*glottis*). At one point the vocal cords are together, thereby blocking the airflow, which builds up *pressure* behind them. Eventually the pressure reaches a level sufficient to force the vocal cords to open and thus allow air to flow through the glottis. Then, the pressure in the glottis falls and, if the tension in the vocal cords is properly adjusted, the reduced pressure allows the cords to come together, and the cycle is repeated. This condition of sustained oscillation occurs for voiced sounds. The *closed-phase* of the oscillation takes place when the glottis is closed and the *volume velocity* is zero. The *open-phase* is characterized by a non-zero volume velocity, in which the lungs and the vocal tract are coupled.

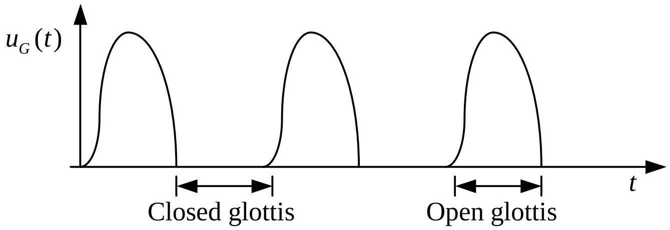

Figure 6.8 Glottal excitation: volume velocity is zero during the closed-phase, during which the vocal cords are closed.

Rosenberg's glottal model [39] defines the shape of the glottal volume velocity with the *open quotient*, or duty cycle, as the ratio of pulse duration to pitch period, and the *speed quotient* as the ratio of the rising to falling pulse durations.

6.2.2. Lossless Tube Concatenation

A widely used model for speech production is based on the assumption that the vocal tract can be represented as a concatenation of lossless tubes, as shown in Figure 6.9. The constant cross-sectional areas $\{A_k\}$ of the tubes approximate the area function $A(x)$ of the vocal tract. If a large number of tubes of short length are used, we reasonably expect the frequency response of the concatenated tubes to be close to those of a tube with continuously varying area function.

For frequencies corresponding to wavelengths that are long compared to the dimensions of the vocal tract, it is reasonable to assume plane wave propagation along the axis of the tubes. If in addition we assume that there are no losses due to viscosity or thermal conduction, and that the area A does not change over time, the sound waves in the tube satisfy the following pair of differential equations:

$$
\begin{aligned}
-\frac{\partial p(x,t)}{\partial x} &= \frac{\rho}{A}\frac{\partial u(x,t)}{\partial t} \\
-\frac{\partial u(x,t)}{\partial x} &= \frac{A}{\rho c^2}\frac{\partial p(x,t)}{\partial t}
\end{aligned}
\tag{6.9}
$$

where $p(x,t)$ is the sound pressure in the tube at position *x* and time *t*, $u(x,t)$ is the volume velocity flow in the tube at position *x* and time *t*, ρ is the density of air in the tube, *c* is the velocity of sound and *A* is the cross-sectional area of the tube.

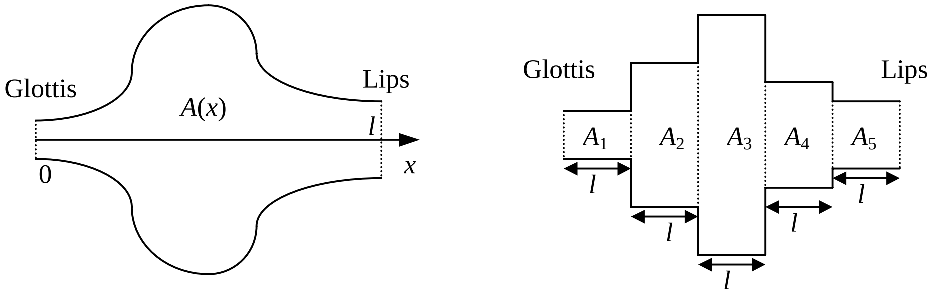

Figure 6.9 Approximation of a tube with continuously varying area *A*(*x*) as a concatenation of 5 lossless acoustic tubes.

Since Eqs. (6.9) are linear, the pressure and volume velocity in tube k^{th} are related by

$$
\begin{aligned}
u_k(x,t) &= u_k^+(t - x/c) - u_k^-(t + x/c) \\
p_k(x,t) &= \frac{\rho c}{A_k}\left[u_k^+(t - x/c) + u_k^-(t + x/c)\right]
\end{aligned}
\tag{6.10}
$$

where $u_k^+(t - x/c)$ and $u_k^-(t - x/c)$ are the traveling waves in the positive and negative directions respectively and *x* is the distance measured from the left-hand end of tube k^{th}: $0 \le x \le l$. The reader can prove that this is indeed the solution by substituting Eq. (6.10) into (6.9).

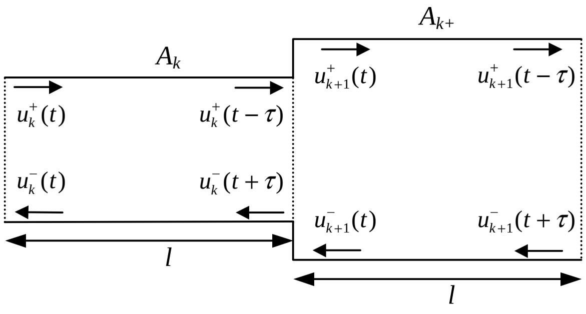

Figure 6.10 Junction between two lossless tubes.

When there is a junction between two tubes, as in Figure 6.10, part of the wave is reflected at the junction, as measured by r_k, the reflection coefficient

$$r_k = \frac{A_{k+1} - A_k}{A_{k+1} + A_k} \tag{6.11}$$

so that the larger the difference between the areas the more energy is reflected. The proof [9] is beyond the scope of this book. Since A_k and A_{k+1} are positive, it is easy to show that r_k satisfies the condition

$$-1 \le r_k \le 1 \tag{6.12}$$

A relationship between the *z*-transforms of the volume velocity at the glottis $u_G[n]$ and the lips $u_L[n]$ for a concatenation of *N* lossless tubes can be derived [9] using a discrete-time version of Eq. (6.10) and taking into account boundary conditions for every junction:

$$V(z) = \frac{U_L(z)}{U_G(z)} = \frac{0.5 z^{-N/2}\left(1 + r_G\right)\prod_{k=1}^{N}\left(1 + r_k\right)}{\begin{bmatrix} 1 & -r_G \end{bmatrix}\left(\prod_{k=1}^{N}\begin{bmatrix} 1 & -r_k \\ -r_k z^{-1} & z^{-1} \end{bmatrix}\right)\begin{bmatrix} 1 \\ 0 \end{bmatrix}} \tag{6.13}$$

where r_G is the reflection coefficient at the glottis and $r_N = r_L$ is the reflection coefficient at the lips. Equation (6.11) is still valid for the glottis and lips, where $A_0 = \rho c / Z_G$ is the equivalent area at the glottis and $A_{N+1} = \rho c / Z_L$ the equivalent area at the lips. Z_G and Z_L are the equivalent impedances at the glottis and lips, respectively. Such impedances relate the volume velocity and pressure, for the lips the expression is

$$U_L(z) = P_L(z) / Z_L \tag{6.14}$$

In general, the concatenation of *N* lossless tubes results in an *N*-pole system as shown in Eq. (6.13). For a concatenation of *N* tubes, there are at most *N*/2 complex conjugate poles, or resonances or formants. These resonances occur when a given frequency gets *trapped* in the vocal tract because it is reflected back at the lips and then again back at the glottis.

Since each tube has length *l* and there are *N* of them, the total length is $L = lN$. The propagation delay in each tube $\tau = l / c$, and the sampling period is $T = 2\tau$, the round trip in a tube. We can find a relationship between the number of tubes *N* and the sampling frequency $F_s = 1/T$:

$$N = \frac{2LF_s}{c} \tag{6.15}$$

For example, for F_s = 8000 kHz, c = 34000 cm/s, and L = 17 cm, the average length of a male adult vocal tract, we obtain N = 8, or alternatively 4 formants. Experimentally, the vocal tract transfer function has been observed to have approximately 1 formant per kilohertz. Shorter vocal tract lengths (females or children) have fewer resonances per kilohertz and vice versa.

The pressure at the lips has been found to approximate the derivative of volume velocity, particularly at low frequencies. Thus, $Z_L(z)$ can be approximated by

$$Z_L(z) \approx R_0(1-z^{-1}) \tag{6.16}$$

which is 0 for low frequencies and reaches R_0 asymptotically. This dependency upon frequency results in a reflection coefficient that is also a function of frequency. For low frequencies, $r_L = 1$, and no loss occurs. At higher frequencies, loss by radiation translates into widening of formant bandwidths.

Similarly, the glottal impedance is also a function of frequency in practice. At high frequencies, Z_G is large and $r_G \approx 1$ so that all the energy is transmitted. For low frequencies, $r_G < 1$, whose main effect is an increase of bandwidth for the lower formants.

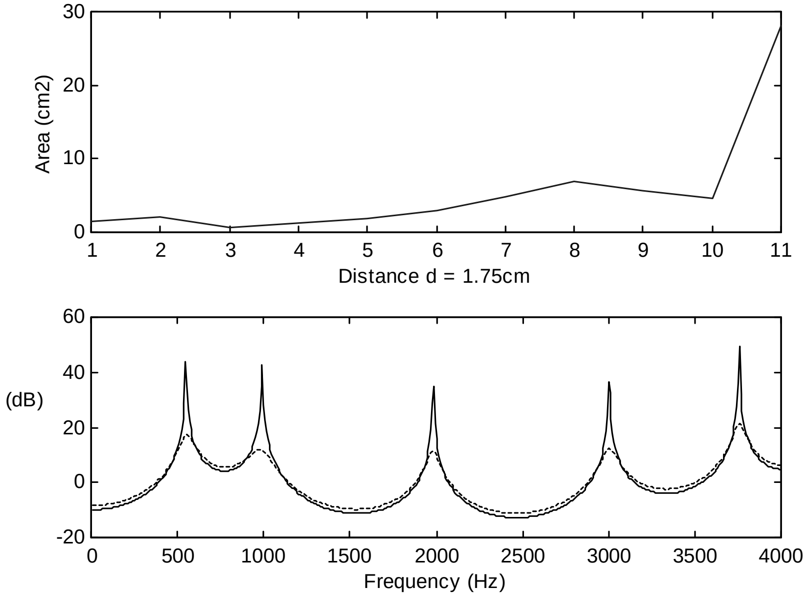

Figure 6.11 Area function and frequency response for vowel /*a*/ and its approximation as a concatenation of 10 lossless tubes. A reflection coefficient at the load of k = 0.72 (dotted line) is displayed. For comparison, the case of k = 1.0 (solid line) is also shown.

Moreover, energy is lost as a result of vibration of the tube walls, which is more pronounced at low frequencies. Energy is also lost, to a lesser extent, as a result of viscous friction between the air and the walls of the tube, particularly at frequencies above 3kHz. The yielding walls tend to raise the resonance frequencies while the viscous and thermal losses tend to lower them. The net effect in the transfer function is a broadening of the resonances' bandwidths.

Despite thermal losses, yielding walls in the vocal tract, and the fact that both r_L and r_G are functions of frequency, the all-pole model of Eq. (6.13) for $V(z)$ has been found to be a good approximation in practice [13]. In Figure 6.11 we show the measured area function of a vowel and its corresponding frequency response obtained using the approximation as a concatenation of 10 lossless tubes with a constant r_L. The measured formants and corresponding bandwidths match quite well with this model despite all the approximations made. Thus, this concatenation of lossless tubes model represents reasonably well the acoustics inside the vocal tract. Inspired by the above results, we describe in Section 6.3 "Linear Predictive Coding," an all-pole model for speech.

In the production of the nasal consonants, the velum is lowered to trap the nasal tract to the pharynx, whereas a complete closure is formed in the oral tract (/m/ at the lips, /n/ just back of the teeth and /ng/ just forward of the velum itself. This configuration is shown in Figure 6.12, which shows two branches, one of them completely closed. For nasals, the radiation occurs primarily at the nostrils. The set of resonances is determined by the shape and length of the three tubes. At certain frequencies, the wave reflected in the closure cancels the wave at the pharynx, preventing energy from appearing at nostrils. The result is that for nasal sounds, the vocal tract transfer function $V(z)$ has anti-resonances (zeros) in addition to resonances. It has also been observed that nasal resonances have broader bandwidths than non-nasal voiced sounds, due to the greater viscous friction and thermal loss because of the large surface area of the nasal cavity.

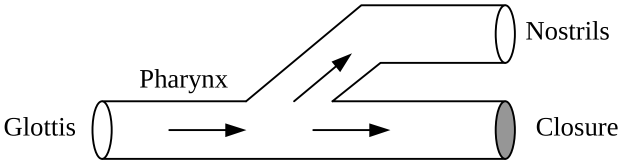

Figure 6.12 Coupling of the nasal cavity with the oral cavity.

6.2.3. Source-Filter Models of Speech Production

As shown in Chapter 10, speech signals are captured by microphones that respond to changes in air pressure. Thus, it is of interest to compute the pressure at the lips $P_L(z)$, which can be obtained as

$$P_L(z) = U_L(z)Z_L(z) = U_G(z)V(z)Z_L(z) \tag{6.17}$$

For voiced sounds we can model $u_G[n]$ as an impulse train convolved with $g[n]$, the glottal pulse (see Figure 6.13). Since $g[n]$ is of finite length, its z-transform is an all-zero system.

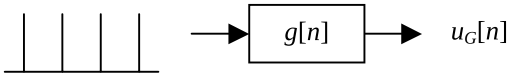

Figure 6.13 Model of the glottal excitation for voiced sounds.

The complete model for both voiced and unvoiced sounds is shown in Figure 6.14. We have modeled $u_G[n]$ in unvoiced sounds as random noise.

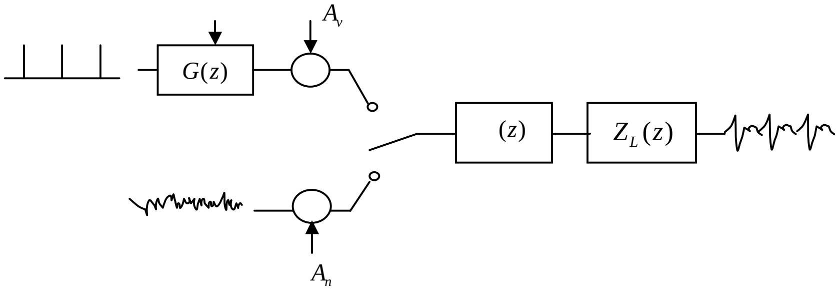

Figure 6.14 General discrete-time model of speech production. The excitation can be either an impulse train with period T and amplitude A_v driving a filter $G(z)$ or random noise with amplitude A_n.

We can simplify the model in Figure 6.14 by grouping $G(z)$, $V(z)$, and $Z_L(z)$ into $H(z)$ for voiced sounds, and $V(z)$ and $Z_L(z)$ into $H(z)$ for unvoiced sounds. The simplified model is shown in Figure 6.15, where we make explicit the fact that the filter changes over time.

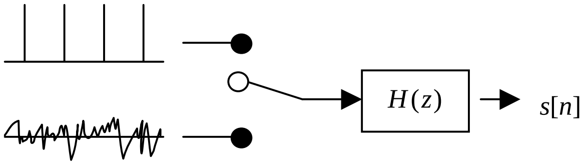

Figure 6.15 Source-filter model for voiced and unvoiced speech.

This model is a decent approximation, but fails on voiced fricatives, since those sounds contain both a periodic component and an aspirated component. In this case, a *mixed excitation* model can be applied, using for voiced sounds a sum of both an impulse train and colored noise (Figure 6.16).

The model in Figure 6.15 is appealing because the source is white (has a flat spectrum) and all the *coloring* is in the filter. Other source-filter decompositions attempt to model the source as the signal at the glottis, in which the source is definitely not white. Since $G(z)$, $Z_L(z)$ contain zeros, and $V(z)$ can also contain zeros for nasals, $H(z)$ is no longer all-pole. However, recall from in Chapter 5, we state that the *z*-transform of $x[n] = a^n u[n]$ is

$$X(z) = \sum_{n=0}^{\infty} a^n z^{-n} = \frac{1}{1-az^{-1}} \qquad \text{for} \qquad |a| < |z| \tag{6.18}$$

so that by inverting Eq. (6.18) we see that a zero can be expressed with infinite poles. This is the reason why all-pole models are still reasonable approximations as long as a large enough number of poles is used. Fant [12] showed that on the average the speech spectrum contains one pole per kHz. Setting the number of poles *p* to $F_s + 2$, where F_s is the sampling frequency expressed in kHz, has been found to work well in practice.

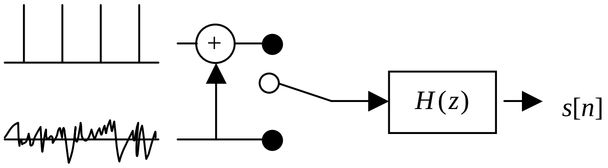

Figure 6.16 A mixed excitation source-filter model of speech.

6.3. LINEAR PREDICTIVE CODING

A very powerful method for speech analysis is based on *linear predictive coding* (LPC) [4, 7, 19, 24, 27], also known as LPC analysis or *auto-regressive* (AR) modeling. This method is widely used because it is fast and simple, yet an effective way of estimating the main parameters of speech signals.

As shown in Section 6.2, an all-pole filter with a sufficient number of poles is a good approximation for speech signals. Thus, we could model the filter *H*(*z*) in Figure 6.15 as

$$H(z) = \frac{X(z)}{E(z)} = \frac{1}{1-\sum_{k=1}^{p} a_k z^{-k}} = \frac{1}{A(z)} \tag{6.19}$$

where *p* is the order of the LPC analysis. The *inverse filter A*(*z*) is defined as

$$A(z) = 1 - \sum_{k=1}^{p} a_k z^{-k} \tag{6.20}$$

Taking inverse *z*-transforms in Eq. (6.19) results in

$$x[n] = \sum_{k=1}^{p} a_k x[n-k] + e[n] \tag{6.21}$$

Linear predictive coding gets its name from the fact that it predicts the current sample as a linear combination of its past *p* samples:

$$\tilde{x}[n] = \sum_{k=1}^{p} a_k x[n-k] \tag{6.22}$$

The prediction error when using this approximation is

$$e[n] = x[n] - \tilde{x}[n] = x[n] - \sum_{k=1}^{p} a_k x[n-k] \tag{6.23}$$

6.3.1. The Orthogonality Principle

To estimate the predictor coefficients from a set of speech samples, we use the short-term analysis technique. Let's define $x_m[n]$ as a segment of speech selected in the vicinity of sample *m*:

$$x_m[n] = x[m+n] \tag{6.24}$$

We define the short-term prediction error for that segment as

$$E_m = \sum_n e_m^2[n] = \sum_n \left(x_m[n] - \tilde{x}_m[n] \right)^2 = \sum_n \left(x_m[n] - \sum_{j=1}^{p} a_j x_m[n-j] \right)^2 \tag{6.25}$$

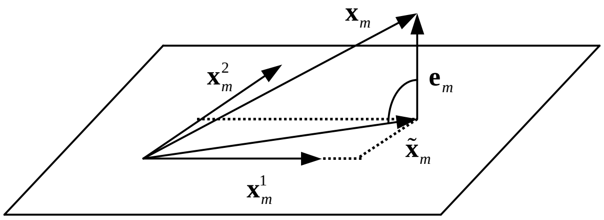

Figure 6.17 The orthogonality principle. The prediction error is orthogonal to the past samples.

In the absence of knowledge about the probability distribution of a_i, a reasonable estimation criterion is minimum mean squared error, introduced in Chapter 4. Thus, given a signal $x_m[n]$, we estimate its corresponding LPC coefficients as those that minimize the total prediction error E_m. Taking the derivative of Eq. (6.25) with respect to a_i and equating to 0, we obtain:

$$< \mathbf{e}_m, \mathbf{x}_m^i >= \sum_n e_m[n] x_m[n-i] = 0 \qquad 1 \le i \le p \tag{6.26}$$

where we have defined $\mathbf{e}_m$ and $\mathbf{x}_m^i$ as vectors of samples, and their inner product has to be 0. This condition, known as *orthogonality principle*, says that the predictor coefficients that minimize the prediction error are such that the error must be orthogonal to the past vectors, and is seen in Figure 6.17.

Equation (6.26) can be expressed as a set of p linear equations

$$\sum_n x_m[n-i] x_m[n] = \sum_{j=1}^{p} a_j \sum_n x_m[n-i] x_m[n-j] \qquad i = 1, 2, \dots, p \tag{6.27}$$

For convenience, we can define the correlation coefficients as

$$\phi_m[i, j] = \sum_n x_m[n-i] x_m[n-j] \tag{6.28}$$

so that Eqs. (6.27) and (6.28) can be combined to obtain the so-called *Yule-Walker* equations:

$$\sum_{j=1}^{p} a_j \phi_m[i, j] = \phi_m[i, 0] \qquad i = 1, 2, \dots, p \tag{6.29}$$

Solution of the set of p linear equations results in the p LPC coefficients that minimize the prediction error. With a_i satisfying Eq. (6.29), the total prediction error in Eq. (6.25) takes on the following value:

$$E_m = \sum_n x_m^2[n] - \sum_{j=1}^{p} a_j \sum_n x_m[n] x_m[n-j] = \phi[0, 0] - \sum_{j=1}^{p} a_j \phi[0, j] \tag{6.30}$$

It is convenient to define a normalized prediction error $u[n]$ with unity energy

$$\sum_n u_m^2[n] = 1 \tag{6.31}$$

and a gain G, such that

$$e_m[n] = G u_m[n] \tag{6.32}$$

The gain G can be computed from the short-term prediction error

$$E_m = \sum_n e_m^2[n] = G^2 \sum_n u_m^2[n] = G^2 \tag{6.33}$$

6.3.2. Solution of the LPC Equations

The solution of the Yule-Walker equations in Eq. (6.29) can be achieved with any standard matrix inversion package. Because of the special form of the matrix here, some efficient solutions are possible, as described below. Also, each solution offers a different insight so we present three different algorithms: the covariance method, the autocorrelation method, and the lattice method.

6.3.2.1. Covariance Method

The covariance method [4] is derived by defining directly the interval over which the summation in Eq. (6.28) takes place:

$$E_m = \sum_{n=0}^{N-1} e_m^2[n] \tag{6.34}$$

so that $\phi_m[i, j]$ in Eq. (6.28) becomes

$$\phi_m[i, j] = \sum_{n=0}^{N-1} x_m[n-i]x_m[n-j] = \sum_{n=-i}^{N-1-j} x_m[n]x_m[n+i-j] = \phi_m[j,i] \tag{6.35}$$

and Eq. (6.29) becomes

$$\begin{pmatrix} \phi_m[1,1] & \phi_m[1,2] & \phi_m[1,3] & \cdots & \phi_m[1,p] \\ \phi_m[2,1] & \phi_m[2,2] & \phi_m[2,3] & \cdots & \phi_m[1,p] \\ \phi_m[3,1] & \phi_m[3,2] & \phi_m[3,3] & \cdots & \phi_m[3,p] \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ \phi_m[p,1] & \phi_m[p,2] & \phi_m[p,3] & \cdots & \phi_m[p,p] \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ \cdots \\ a_p \end{pmatrix} = \begin{pmatrix} \phi_m[1,0] \\ \phi_m[2,0] \\ \phi_m[3,0] \\ \cdots \\ \phi_m[p,0] \end{pmatrix} \tag{6.36}$$

which can be expressed as the following matrix equation

$$\Phi \mathbf{a} = \psi \tag{6.37}$$

where the matrix Φ in Eq. (6.37) is symmetric and *positive definite*, for which efficient methods are available, such as the Cholesky decomposition. For this method, also called the squared root method, the matrix Φ is expressed as

$$\Phi = \mathbf{V}\mathbf{D}\mathbf{V}^t \tag{6.38}$$

where $\mathbf{V}$ is a lower triangular matrix (whose main diagonal elements are 1's), and $\mathbf{D}$ is a diagonal matrix. So each element of Φ can be expressed as

$$\phi[i, j] = \sum_{k=1}^{j} V_{ik} d_k V_{jk} \qquad 1 \le j < i \tag{6.39}$$

or alternatively

$$V_{ij}d_j = \phi[i, j] - \sum_{k=1}^{j-1} V_{ik} d_k V_{jk} \qquad 1 \le j < i \tag{6.40}$$

and for the diagonal elements

$$\phi[i,i] = \sum_{k=1}^{i} V_{ik} d_k V_{ik} \tag{6.41}$$

or alternatively

$$d_i = \phi[i,i] - \sum_{k=1}^{i-1} V_{ik}^2 d_k \ , \qquad i \ge 2 \tag{6.42}$$

with

$$d_1 = \phi[1,1] \tag{6.43}$$

The Cholesky decomposition starts with Eq. (6.43) then alternates between Eqs. (6.40) and (6.42). Once the matrices $\mathbf{V}$ and $\mathbf{D}$ have been determined, the LPC coefficients are solved in a two-step process. The combination of Eqs. (6.37) and (6.38) can be expressed as

$$\mathbf{VY} = \psi \tag{6.44}$$

with

$$\mathbf{Y} = \mathbf{DV}^t\mathbf{a} \tag{6.45}$$

or alternatively

$$\mathbf{V}^t\mathbf{a} = \mathbf{D}^{-1}\mathbf{Y} \tag{6.46}$$

Therefore, given matrix $\mathbf{V}$ and Eq. (6.44), $\mathbf{Y}$ can be solved recursively as

$$Y_i = \psi_i - \sum_{j=1}^{i-1} V_{ij} Y_j \ , \qquad 2 \le i \le p \tag{6.47}$$

with the initial condition

$$Y_1 = \psi_1 \tag{6.48}$$

Having determined $\mathbf{Y}$, Eq. (6.46) can be solved recursively in a similar way

$$a_i = Y_i / d_i - \sum_{j=i+1}^{p} V_{ji} a_j \ , \qquad 1 \le i < p \tag{6.49}$$

with the initial condition

$$a_p = Y_p / d_p \tag{6.50}$$

where the index i in Eq. (6.49) proceeds backwards.

The term covariance analysis is somewhat of a misnomer, since we know from Chapter 5 that the covariance of a signal is the correlation of that signal with its mean removed. It was so called because the matrix in Eq. (6.36) has the properties of a covariance matrix, though this algorithm is more like a cross-correlation.

6.3.2.2. Autocorrelation Method

The summation in Eq. (6.28) had no specific range. In the autocorrelation method [24, 27], we assume that $x_m[n]$ is 0 outside the interval $0 \le n < N$:

$$x_m[n] = x[m+n]w[n] \tag{6.51}$$

with $w[n]$ being a window (such as a Hamming window) which is 0 outside the interval $0 \le n < N$. With this assumption, the corresponding prediction error $e_m[n]$ is non-zero over the interval $0 \le n < N + p$, and, therefore, the total prediction error takes on the value

$$E_m = \sum_{n=0}^{N+p-1} e_m^2[n] \tag{6.52}$$

With this range, Eq. (6.28) can be expressed as

$$\phi_m[i, j] = \sum_{n=0}^{N+p-1} x_m[n-i]x_m[n-j] = \sum_{n=0}^{N-1-(i-j)} x_m[n]x_m[n+i-j] \tag{6.53}$$

or alternatively

$$\phi_m[i, j] = R_m[i-j] \tag{6.54}$$

with $R_m[k]$ being the autocorrelation sequence of $x_m[n]$:

$$R_m[k] = \sum_{n=0}^{N-1-k} x_m[n]x_m[n+k] \tag{6.55}$$

Combining Eqs. (6.54) and (6.29), we obtain

$$\sum_{j=1}^{p} a_j R_m[|i-j|] = R_m[i] \tag{6.56}$$

which corresponds to the following matrix equation

$$\begin{pmatrix} R_m[0] & R_m[1] & R_m[2] & \cdots & R_m[p-1] \\ R_m[1] & R_m[0] & R_m[1] & \cdots & R_m[p-2] \\ R_m[2] & R_m[1] & R_m[0] & \cdots & R_m[p-3] \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ R_m[p-1] & R_m[p-2] & R_m[p-3] & \cdots & R_m[0] \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ \cdots \\ a_p \end{pmatrix} = \begin{pmatrix} R_m[1] \\ R_m[2] \\ R_m[3] \\ \cdots \\ R_m[p] \end{pmatrix} \tag{6.57}$$

The matrix in Eq. (6.57) is symmetric and all the elements in its diagonals are identical. Such matrices are called *Toeplitz*. Durbin's recursion exploits this fact resulting in a very efficient algorithm (for convenience, we omit the subscript m of the autocorrelation function), whose proof is beyond the scope of this book:

1. Initialization

$$E^0 = R[0] \tag{6.58}$$

2. Iteration. For $i = 1, \cdots, p$ do the following recursion:

$$k_i = \left(R[i] - \sum_{j=1}^{i-1} a_j^{i-1} R[i-j] \right) / E^{i-1} \tag{6.59}$$

$$a_i^i = k_i \tag{6.60}$$

$$a_j^i = a_j^{i-1} - k_i a_{i-j}^{i-1}, \qquad 1 \le j < i \tag{6.61}$$

$$E^i = (1 - k_i^2) E^{i-1} \tag{6.62}$$

3. Final solution:

$$a_j = a_j^p \qquad 1 \le j \le p \tag{6.63}$$

where the coefficients k_i, called *reflection coefficients*, are bounded between –1 and 1 (see Section 6.3.2.3). In the process of computing the predictor coefficients of order p, the recursion finds the solution of the predictor coefficients for all orders less than p.

Replacing $R[j]$ by the normalized autocorrelation coefficients $r[j]$, defined as

$$r[j] = R[j] / R[0] \tag{6.64}$$

results in identical LPC coefficients, and the recursion is more robust to problems with arithmetic precision. Likewise, the normalized prediction error at iteration i is defined by dividing Eq. (6.30) by $R[0]$, which, using Eq. (6.54), results in

$$V^i = \frac{E^i}{R[0]} = 1 - \sum_{j=1}^{i} a_j r[j] \tag{6.65}$$

The normalized prediction error is, using Eqs. (6.62) and (6.65),

$$V^p = \prod_{i=1}^{p} (1 - k_i^2) \tag{6.66}$$

6.3.2.3. Lattice Formulation

In this section we derive the lattice formulation [7, 19], an equivalent algorithm to the Levinson Durbin recursion, which has some precision benefits. It is advantageous to define the *forward prediction error* obtained at stage *i* of the Levinson Durbin procedure as

$$e^i[n] = x[n] - \sum_{k=1}^{i} a_k^i x[n-k] \tag{6.67}$$

whose *z*-transform is given by

$$E^i(z) = A^i(z) X(z) \tag{6.68}$$

with $A^i(z)$ being defined by

$$A^i(z) = 1 - \sum_{k=1}^{i} a_k^i z^{-k} \tag{6.69}$$

which, combined with Eq. (6.61), results in the following recursion:

$$A^i(z) = A^{i-1}(z) - k_i z^{-i} A^{i-1}(z^{-1}) \tag{6.70}$$

Similarly, we can define the so-called *backward prediction error* as

$$b^i[n] = x[n-i] - \sum_{k=1}^{i} a_k^i x[n+k-i] \tag{6.71}$$

whose *z*-transform is

$$B^i(z) = z^{-i} A^i(z^{-1}) X(z) \tag{6.72}$$

Now combining Eqs. (6.68), (6.70), and (6.72), we obtain

$$E^i(z) = A^{i-1}(z) X(z) - k_i z^{-i} A^{i-1}(z^{-1}) X(z) = E^{i-1}(z) - k_i B^{i-1}(z) \tag{6.73}$$

whose inverse *z*-transform is given by

$$e^i[n] = e^{i-1}[n] - k_i b^{i-1}[n-1] \tag{6.74}$$

Also, substituting Eqs. (6.70) into (6.72) and using Eq. (6.68), we obtain

$$B^i(z) = z^{-1} B^{i-1}(z) - k_i E^{i-1}(z) \tag{6.75}$$

whose inverse z-transform is given by

$$b^i[n] = b^{i-1}[n-1] - k_i e^{i-1}[n] \tag{6.76}$$

Equations (6.74) and (6.76) define the forward and backward prediction error sequences for an i^{th}-order predictor in terms of the corresponding forward and backward prediction errors of an $(i - 1)^{th}$-order predictor. We initialize the recursive algorithm by noting that the 0^{th}-order predictor is equivalent to using no predictor at all; thus

$$e^0[n] = b^0[n] = x[n] \tag{6.77}$$

and the final prediction error is $e[n] = e^p[n]$.

A block diagram of the lattice method is given in Figure 6.18, which resembles a lattice, whence its name.

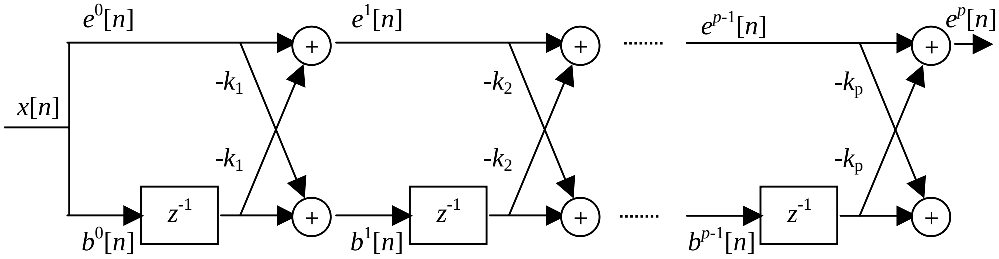

Figure 6.18 Block diagram of the lattice filter.

While the computation of the k_i coefficients can be done through the Levinson Durbin recursion of Eqs. (6.59) through (6.62), it can be shown that an equivalent calculation can be found as a function of the forward and backward prediction errors. To do so we minimize the sum of the forward prediction errors

$$E^i = \sum_{n=0}^{N-1} \left(e^i[n]\right)^2 \tag{6.78}$$

by substituting Eq. (6.74) in (6.78), taking the derivative with respect to k_i, and equating to 0:

$$k_i = \frac{\sum_{n=0}^{N-1} e^{i-1}[n] b^{i-1}[n-1]}{\sum_{n=0}^{N-1} \left(b^{i-1}[n-1]\right)^2} \tag{6.79}$$

Using Eqs. (6.67) and (6.71), it can be shown that

$$\sum_{n=0}^{N-1}\left(e^{i-1}[n]\right)^2 = \sum_{n=0}^{N-1}\left(b^{i-1}[n-1]\right)^2 \tag{6.80}$$

since minimization of both yields identical Yule-Walker equations. Thus Eq. (6.79) can be alternatively expressed as

$$k_i = \frac{\sum_{n=0}^{N-1} e^{i-1}[n]b^{i-1}[n-1]}{\sqrt{\sum_{n=0}^{N-1}\left(e^{i-1}[n]\right)^2 \sum_{n=0}^{N-1}\left(b^{i-1}[n-1]\right)^2}} = \frac{<\mathbf{e}^{i-1},\mathbf{b}^{i-1}>}{\left|\mathbf{e}^{i-1}\right|\left|\mathbf{b}^{i-1}\right|} \tag{6.81}$$

where we have defined the vectors $\mathbf{e}^i = \left(e^i[0]\cdots e^i[N-1]\right)$ and $\mathbf{b}^i = \left(b^i[0]\cdots b^i[N-1]\right)$. The inner product of two vectors **x** and **y** is defined as

$$<\mathbf{x},\mathbf{y}>= \sum_{n=0}^{N-1} x[n]y[n] \tag{6.82}$$

and its norm as

$$\left|\mathbf{x}\right|^2 =<\mathbf{x},\mathbf{x}>= \sum_{n=0}^{N-1} x^2[n] \tag{6.83}$$

Equation (6.81) has the form of a normalized cross-correlation function, and, therefore, the reason the reflection coefficients are also called *partial correlation coefficients* (PARCOR). As with any normalized cross-correlation function, the k_i coefficients are bounded by

$$-1 \le k_i \le 1 \tag{6.84}$$

This is a necessary and sufficient condition for all the roots of the polynomial $A(z)$ to be inside the unit circle, therefore guaranteeing a stable filter. This condition can be checked to avoid numerical imprecision by stopping the recursion if the condition is not met. The inverse lattice filter can be seen in Figure 6.19, which resembles the lossless tube model. This is why the k_i are also called *reflection coefficients*.

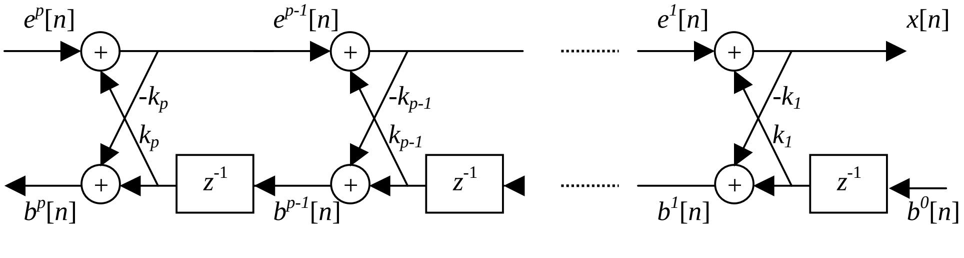

Figure 6.19 Inverse lattice filter used to generate the speech signal, given its residual.

Lattice filters are often used in fixed-point implementation, because lack of precision doesn't result in unstable filters. Any error that may take place – for example due to quantization – is generally not be sufficient to cause k_i to fall outside the range in Eq. (6.84). If, owing to round-off error, the reflection coefficient falls outside the range, the lattice filter can be ended at the previous step.

More importantly, linearly varying coefficients can be implemented in this fashion. While, typically, the reflection coefficients are constant during the analysis frame, we can implement a linear interpolation of the reflection coefficients to obtain the error signal. If the coefficients of both frames are in the range in Eq. (6.84), the linearly interpolated reflection coefficients also have that property, and thus the filter is stable. This is a property that the predictor coefficients don't have.

6.3.3. Spectral Analysis via LPC

Let's now analyze the frequency-domain behavior of the LPC analysis by evaluating

$$H(e^{j\omega}) = \frac{G}{1 - \sum_{k=1}^{p} a_k e^{-j\omega k}} = \frac{G}{A(e^{j\omega})} \tag{6.85}$$

which is an *all-pole* or IIR filter. If we plot $H(e^{j\omega})$, we expect to see peaks at the roots of the denominator. Figure 6.20 shows the 14-order LPC spectrum of the vowel of Figure 6.3 (d).

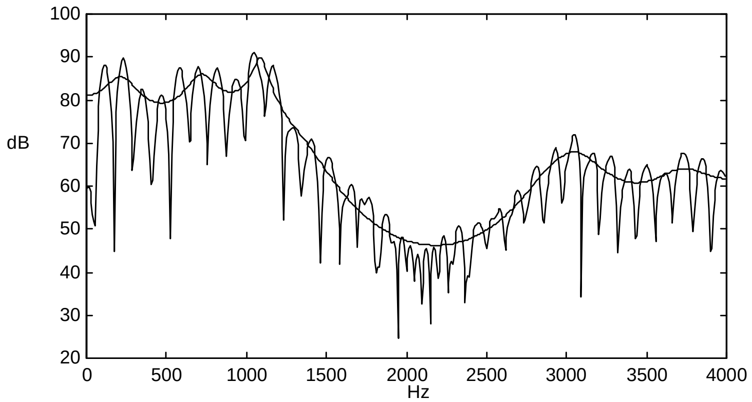

Figure 6.20 LPC spectrum of the */ah/* phoneme in the word ***lifes*** of Figure 6.3. Used here are a 30-ms Hamming window and the autocorrelation method with *p* = 14. The short-time spectrum is also shown.

For the autocorrelation method, the squared error of Eq. (6.52) can be expressed, using Eq. (6.85) and Parseval's theorem, as

$$E_m = \frac{G^2}{2\pi}\int_{-\pi}^{\pi}\frac{|X_m(e^{j\omega})|^2}{|H(e^{j\omega})|^2}d\omega \tag{6.86}$$

Since the integrand in Eq. (6.86) is positive, minimizing E_m is equivalent to minimizing the ratio of the energy spectrum of the speech segment $|X_m(e^{j\omega})|^2$ to the magnitude squared of the frequency response of the linear system $|H(e^{j\omega})|^2$. The LPC spectrum matches more closely the peaks than the valleys (see Figure 6.20), because the regions where $|X_m(e^{j\omega})|>|H(e^{j\omega})|$ contribute more to the error than those where $|H(e^{j\omega})|>|X_m(e^{j\omega})|$.

Even nasals, which have zeros in addition to poles, can be represented with an infinite number of poles. In practice, if *p* is large enough we can approximate the signal spectrum with arbitrarily small error. Figure 6.21 shows different fits for different values of *p*. The higher *p*, the more details of the spectrum are preserved.

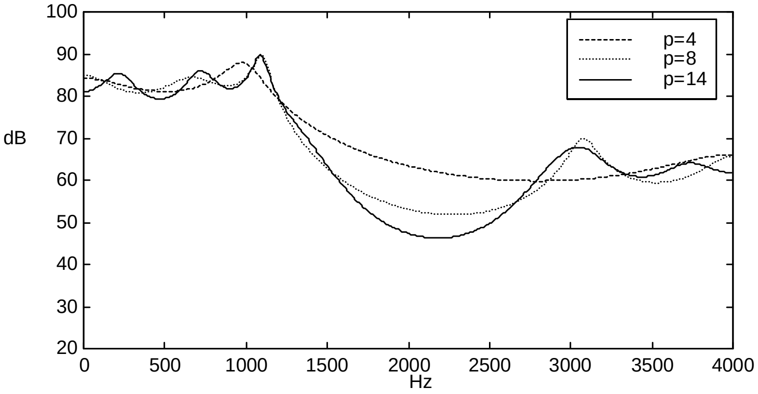

Figure 6.21 LPC spectra of Figure 6.20 for various values of the predictor order *p*.

The prediction order is not known for arbitrary speech, so we need to set it to balance spectral detail with estimation errors.

6.3.4. The Prediction Error

So far, we have concentrated on the filter component of the source-filter model. Using Eq. (6.23), we can compute the prediction error signal, also called the *excitation*, or *residual* signal. For unvoiced speech synthetically generated by white noise following an LPC filter we expect the residual to be approximately white noise. In practice, this approximation is quite good, and replacement of the residual by white noise followed by the LPC filter typically results in no audible difference. For voiced speech synthetically generated by an impulse train following an LPC filter, we expect the residual to approximate an impulse train. In practice, this is not the case, because the all-pole assumption is not altogether valid; thus, the residual, although it contains spikes, is far from an impulse train. Replacing the residual by an impulse train, followed by the LPC filter, results in speech that sounds somewhat robotic, partly because real speech is not perfectly periodic (it has a random component as well), and because the zeroes are not modeled with the LPC filter. Residual signals computed from inverse LPC filters for several vowels are shown in Figure 6.22.

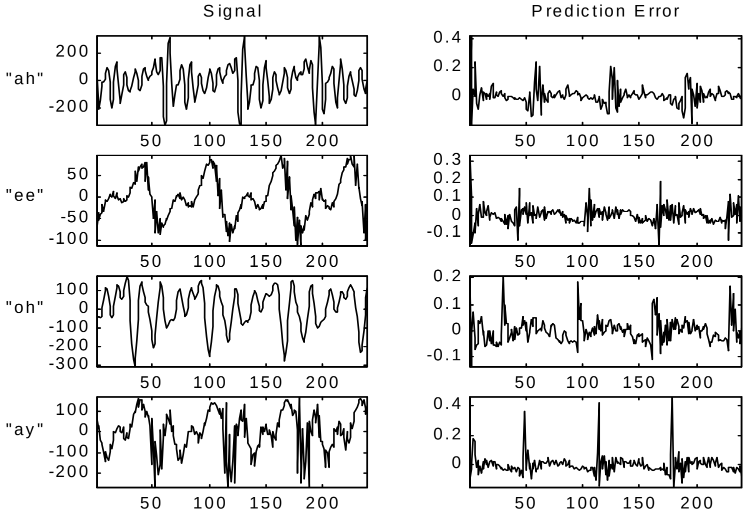

Figure 6.22 LPC prediction error signals for several vowels.

How do we choose *p*? This is an important design question. Larger values of *p* lead to lower prediction errors (see Figure 6.23). Unvoiced speech has higher error than voiced speech, because the LPC model is more accurate for voiced speech. In general, the normalized error rapidly decreases, and then converges to a value of around 12 - 14 for 8 kHz speech. If we use a large value of *p*, we are fitting the individual harmonics; thus the LPC filter is modeling the source, and the separation between source and filter is not going to be so good. The more coefficients we have to estimate, the larger the variance of their estimates, since the number of available samples is the same. A rule of thumb is to use 1 complex pole per kHz plus 2 - 4 poles to model the radiation and glottal effects.

For unvoiced speech, both the autocorrelation and the covariance methods provide similar results. For voiced speech, however, the covariance method can provide better estimates if the analysis window is shorter than the local pitch period and the window only includes samples from the closed phase (when the vocal tract is closed at the glottis and speech signal is due mainly to free resonances). This is called *pitch synchronous* analysis and results in lower prediction error, because the true excitation is close to zero during the whole analysis window. During the open phase, the trachea, the vocal folds, and the vocal tract are acoustically coupled, and this coupling will change the free resonances. Additionally, the prediction error is higher for both the autocorrelation and the covariance methods if samples from the open phase are included in the analysis window, because the prediction during those instants is poor.

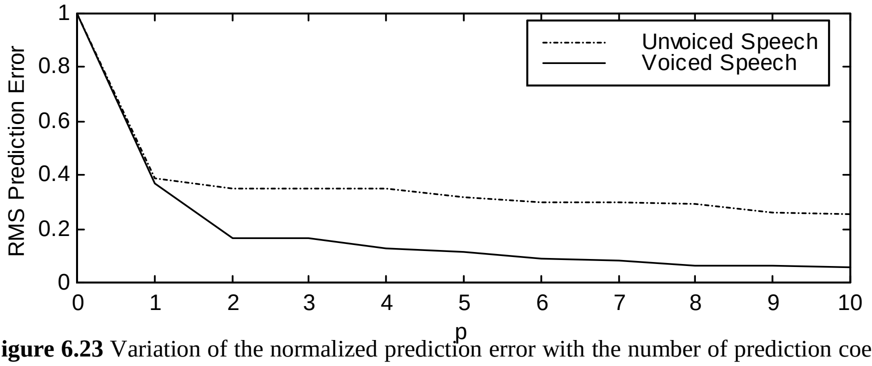

Figure 6.23 Variation of the normalized prediction error with the number of prediction coefficients p for the voiced segment of Figure 6.3 and the unvoiced speech of Figure 6.5. The autocorrelation method was used with a 30 ms Hamming window, and a sampling rate of 8 kHz.

6.3.5. Equivalent Representations

There are a number of alternate useful representations of the predictor coefficients. The most important are the line spectrum pairs, reflection coefficients, log-area ratios, and the roots of the predictor polynomial.

6.3.5.1. Line Spectral Frequencies

Line Spectral Frequencies (LSF) [18] provide an equivalent representation of the predictor coefficients that is very popular in speech coding. It is derived from computing the roots of the polynomials $P(z)$ and $Q(z)$ defined as

$$P(z) = A(z) + z^{-(p+1)} A(z^{-1}) \tag{6.87}$$

$$Q(z) = A(z) - z^{-(p+1)} A(z^{-1}) \tag{6.88}$$

To gain insight on these roots, look at a second-order predictor filter with a pair of complex roots:

$$A(z) = 1 - a_1 z^{-1} - a_2 z^{-2} = 1 - 2\rho_0 \cos(2\pi f_0) z^{-1} + \rho_0^2 z^{-2} \tag{6.89}$$

where $0 < \rho_0 < 1$ and $0 < f_0 < 0.5$. Inserting Eq. (6.89) into (6.87) and (6.88) results in

$$\begin{aligned} P(z) &= 1 - (a_1 + a_2) z^{-1} - (a_1 + a_2) z^{-2} + z^{-3} \\ Q(z) &= 1 - (a_1 - a_2) z^{-1} + (a_1 - a_2) z^{-2} - z^{-3} \end{aligned} \tag{6.90}$$

From Eq. (6.90) we see that $z = -1$ is a root of $P(z)$ and $z = 1$ a root of $Q(z)$, which can be divided out and results in

$$\begin{aligned} P(z) &= (1+z^{-1})(1-2\beta_1 z^{-1}+z^{-2}) \\ Q(z) &= (1-z^{-1})(1-2\beta_2 z^{-1}+z^{-2}) \end{aligned} \tag{6.91}$$

where β_1 and β_2 are given by

$$\begin{aligned} \beta_1 &= \frac{a_1+a_2+1}{2} = \rho_0\cos(2\pi f_0)+\frac{1-\rho_0^2}{2} \\ \beta_2 &= \frac{a_1-a_2-1}{2} = \rho_0\cos(2\pi f_0)-\frac{1-\rho_0^2}{2} \end{aligned} \tag{6.92}$$

It can be shown that $|\beta_1|<1$ and $|\beta_2|<1$ for all possible values of f_0 and ρ_0. With this property, the roots of *P*(*z*) and *Q*(*z*) in Eq. (6.91) are complex and given by $\beta_1 \pm j\sqrt{1-\beta_1^2}$ and $\beta_2 \pm j\sqrt{1-\beta_2^2}$, respectively. Because they lie in the unit circle, they can be uniquely represented by their angles

$$\begin{aligned} \cos(2\pi f_1) &= \rho_0\cos(2\pi f_0)+\frac{1-\rho_0^2}{2} \\ \cos(2\pi f_2) &= \rho_0\cos(2\pi f_0)-\frac{1-\rho_0^2}{2} \end{aligned} \tag{6.93}$$

where f_1 and f_2 are the *line spectral frequencies* of *A*(*z*). Since $|\rho_0|<1$, $\cos(2\pi f_2)<\cos(2\pi f_0)$, and thus $f_2 > f_0$. It's also the case that $\cos(2\pi f_1)>\cos(2\pi f_0)$ and thus $f_1 < f_0$. Furthermore, as $\rho_0 \to 1$, we see from Eq. (6.93) that $f_1 \to f_0$ and $f_2 \to f_0$. We conclude that, given a pole at f_0, the two line spectral frequencies bracket it, *i.e.*, $f_1 < f_0 < f_2$, and that they are closer together as the pole of the second-order resonator gets closer to the unit circle.

We have proven that for a second-order predictor, the roots of *P*(*z*) and *Q*(*z*) lie in the unit circle, that ± 1 are roots, and that, once sorted, the roots of *P*(*z*) and *Q*(*z*) alternate. Although we do not prove it here, it can be shown that these conclusions hold for other predictor orders, and, therefore, the *p* predictor coefficients can be transformed into *p* line spectral frequencies. We also know that $z=1$ is always a root of *Q*(*z*), whereas $z=-1$ is a root of *P*(*z*) for even *p* and a root of *Q*(*z*) for odd *p*.

To compute the LSF for $p>2$, we replace $z=\cos(\omega)$ and compute the roots of $P(\omega)$ and $Q(\omega)$ by any available root finding method. A popular technique, given that there are *p* roots which are real in ω and bounded between 0 and 0.5, is to bracket them by observing changes in sign of both functions in a dense grid. To compute the predictor coefficients from the LSF coefficients we can factor *P*(*z*) and *Q*(*z*) as a product of second-order filters as in Eq. (6.91), and then $A(z)=\left(P(z)+Q(z)\right)/2$.

In practice, LSF are useful because of *sensitivity* (a quantization of one coefficient generally results in a spectral change only around that frequency) and *efficiency* (LSF result in low spectral distortion). This doesn't occur with other representations. As long as the LSF coefficients are ordered, the resulting LPC filter is stable, though the proof is beyond the scope of this book. LSF coefficients are used extensively in Chapter 7.

6.3.5.2. Reflection Coefficients

For the autocorrelation method, the predictor coefficients may be obtained from the reflection coefficients by the following recursion:

$$\begin{aligned} a_i^i &= k_i & i &= 1, \cdots, p \\ a_j^i &= a_j^{i-1} - k_i a_{i-j}^{i-1} & 1 &\le j < i \end{aligned} \tag{6.94}$$

where $a_i = a_i^p$. Similarly, the reflection coefficients may be obtained from the prediction coefficients using a backward recursion of the form

$$\begin{aligned} k_i &= a_i^i & i &= p, \cdots, 1 \\ a_j^{i-1} &= \frac{a_j^i + a_i^i a_{i-j}^i}{1 - k_i^2} & 1 &\le j < i \end{aligned} \tag{6.95}$$

where we initialize $a_i^p = a_i$.

Reflection coefficients are useful when implementing LPC filters whose values are interpolated over time, because, unlike the predictor coefficients, they are guaranteed to be stable at all times as long as the anchors satisfy Eq. (6.84).

6.3.5.3. Log-Area Ratios

The *log-area ratio* coefficients are defined as

$$g_i = \ln\left(\frac{1-k_i}{1+k_i}\right) \tag{6.96}$$

with the inverse being given by

$$k_i = \frac{1-e^{g_i}}{1+e^{g_i}} \tag{6.97}$$

The log-area ratio coefficients are equal to the natural logarithm of the ratio of the areas of adjacent sections of a lossless tube equivalent of the vocal tract having the same transfer function. Since for stable predictor filters $-1 < k_i < 1$, we have from Eq. (6.96) that $-\infty < g_i < \infty$. For speech signals, it is not uncommon to have some reflection coefficients close to 1, and quantization of those values can cause a large change in the predictor's transfer function. On the other hand, the log-area ratio coefficients have relatively flat spectral sensitivity (i.e., a small change in their values causes a small change in the transfer function) and thus are useful in coding.

6.3.5.4. Roots of Polynomial

An alternative to the predictor coefficients results from computing the complex roots of the predictor polynomial:

$$A(z) = 1 - \sum_{k=1}^{p} a_k z^{-k} = \prod_{k=1}^{p} (1 - z_k z^{-1}) \tag{6.98}$$

These roots can be represented as

$$z_k = e^{(-\pi b_k + j 2\pi f_k)/F_s} \tag{6.99}$$

where b_k, f_k, and F_s represent the bandwidth, center frequency, and sampling frequency, respectively. Since a_k are real, all complex roots occur in conjugate pairs so that if (b_k, f_k) is a root, so is $(b_k, -f_k)$. The bandwidths b_k are always positive, because the roots are inside the unit circle ($|z_k| < 1$) for a stable predictor. Real roots $z_k = e^{-\pi b_k / F_s}$ can also occur. While algorithms exist to compute the complex roots of a polynomial, in practice there are sometimes numerical difficulties in doing so.

If the roots are available, it is straightforward to compute the predictor coefficients by using Eq. (6.98). Since the roots of the predictor polynomial represent resonance frequencies and bandwidths, they are used in formant synthesizers of Chapter 16.

6.4. CEPSTRAL PROCESSING

A *homomorphic* transformation $\hat{x}[n] = D(x[n])$ is a transformation that converts a convolution

$$x[n] = e[n] * h[n] \tag{6.100}$$

into a sum

$$\hat{x}[n] = \hat{e}[n] + \hat{h}[n] \tag{6.101}$$

In this section we introduce the *cepstrum* as one homomorphic transformation [32] that allows us to separate the source from the filter. We show that we can find a value N such that the cepstrum of the filter $\hat{h}[n] \approx 0$ for $n \geq N$, and that the cepstrum of the excitation $\hat{e}[n] \approx 0$ for $n < N$. With this assumption, we can approximately recover both $e[n]$ and $h[n]$ from $\hat{x}[n]$ by homomorphic filtering. In Figure 6.24, we show how to recover $h[n]$ with a homomorphic filter:

$$l[n] = \begin{cases} 1 & |n| < N \\ 0 & |n| \ge N \end{cases} \tag{6.102}$$

where D is the cepstrum operator.

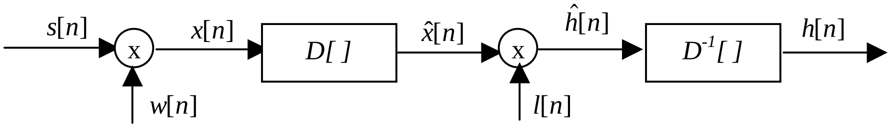

Figure 6.24 Homomorphic filtering to recover the filter's response from a periodic signal. We have used the homomorphic filter of Eq. (6.102).

The excitation signal can similarly recovered with a homomorphic filter given by

$$l[n] = \begin{cases} 1 & |n| \ge N \\ 0 & |n| < N \end{cases} \tag{6.103}$$

6.4.1. The Real and Complex Cepstrum

The *real cepstrum* of a digital signal $x[n]$ is defined as

$$c[n] = \frac{1}{2\pi} \int_{-\pi}^{\pi} \ln |X(e^{j\omega})| \, e^{j\omega n} d\omega \tag{6.104}$$

and the *complex cepstrum* of $x[n]$ is defined as

$$\hat{x}[n] = \frac{1}{2\pi} \int_{-\pi}^{\pi} \ln X(e^{j\omega}) \; e^{j\omega n} d\omega \tag{6.105}$$

where the complex logarithm is used:

$$\hat{X}(e^{j\omega}) = \ln X(e^{j\omega}) = \ln |X(e^{j\omega})| + j\theta(\omega) \tag{6.106}$$

and the phase $\theta(\omega)$ is given by

$$\theta(\omega) = \arg\left[X(e^{j\omega}) \right] \tag{6.107}$$

You can see from Eqs. (6.104) and (6.105) that both the real and the complex cepstrum satisfy Eq. (6.101) and thus they are homomorphic transformations.

If the signal $x[n]$ is real, both the real cepstrum $c[n]$ and the complex cepstrum $\hat{x}[n]$ are also real signals. Therefore the term complex cepstrum doesn't mean that it is a complex signal but rather that the complex logarithm is taken.

It can easily be shown that $c[n]$ is the even part of $\hat{x}[n]$:

$$c[n] = \frac{\hat{x}[n] + \hat{x}[-n]}{2} \tag{6.108}$$

From here on, when we refer to cepstrum without qualifiers, we are referring to the real cepstrum, since it is the most widely used in speech technology.

The cepstrum was invented by Bogert et al. [6], and its term was coined by reversing the first syllable of the word spectrum, given that it is obtained by taking the inverse Fourier transform of the log-spectrum. Similarly, they defined the term *quefrency* to represent the independent variable *n* in *c*[*n*]. The quefrency has dimension of time.

6.4.2. Cepstrum of Pole-Zero Filters

A very general type of filters are those with rational transfer functions

$$H(z) = \frac{Az^r \prod_{k=1}^{M_i}(1 - a_k z^{-1}) \prod_{k=1}^{M_o}(1 - u_k z)}{\prod_{k=1}^{N_i}(1 - b_k z^{-1}) \prod_{k=1}^{N_o}(1 - v_k z)} \tag{6.109}$$

with the magnitudes of a_k, b_k, u_k and v_k all less than 1. Therefore, $(1 - a_k z^{-1})$ and $(1 - b_k z^{-1})$ represent the zeros and poles inside the unit circle, whereas $(1 - u_k z)$ and $(1 - v_k z)$ represent the zeros and poles outside the unit circle, and z^r is a shift from the time origin. Thus, the complex logarithm is

$$\begin{aligned}\hat{H}(z) &= \ln[A] + \ln[z^r] + \sum_{k=1}^{M_i} \ln(1 - a_k z^{-1}) \\ &- \sum_{k=1}^{N_i} \ln(1 - b_k z^{-1}) + \sum_{k=1}^{M_o} \ln(1 - u_k z) - \sum_{k=1}^{N_o} \ln(1 - v_k z)\end{aligned} \tag{6.110}$$

where the term $\log[z^r]$ contributes to the imaginary part of the complex cepstrum only with a term $j\omega r$. Since it just carries information about the time origin, it's typically ignored. We use the Taylor series expansion

$$\ln(1 - x) = -\sum_{n=1}^{\infty} \frac{x^n}{n} \tag{6.111}$$

in Eq. (6.110) and take inverse *z*-transforms to obtain

$$\hat{h}[n] = \begin{cases} \log[A] & n = 0 \\ \sum_{k=1}^{N_i} \frac{b_k^n}{n} - \sum_{k=1}^{M_i} \frac{a_k^n}{n} & n > 0 \\ \sum_{k=1}^{M_o} \frac{u_k^n}{n} - \sum_{k=1}^{N_o} \frac{v_k^n}{n} & n < 0 \end{cases} \tag{6.112}$$

If the filter's impulse response doesn't have zeros or poles outside the unit circle, the so-called *minimum phase* signals, then $\hat{h}[n] = 0$ for $n < 0$. *Maximum phase* signals are those with $\hat{h}[n] = 0$ for $n > 0$. If a signal is minimum phase, its complex cepstrum can be uniquely determined from its real cepstrum:

$$\hat{h}[n] = \begin{cases} 0 & n < 0 \\ c[n] & n = 0 \\ 2c[n] & n > 0 \end{cases} \tag{6.113}$$

It is easy to see from Eq. (6.112) that both the real and complex cepstrum are decaying sequences, which is the reason why, typically, a finite number of coefficients are sufficient to approximate it, and, therefore, people refer to the truncated cepstrum signal as a *cepstrum vector*.

6.4.2.1. LPC-Cepstrum

The case when the rational transfer function in Eq. (6.109) has been obtained with an LPC analysis is particularly interesting, since LPC analysis is such a widely used method. While Eq. (6.112) applies here, too, it is useful to find a recursion which doesn't require us to compute the roots of the predictor polynomial. Given the LPC filter

$$H(z) = \frac{G}{1 - \sum_{k=1}^{p} a_k z^{-k}} \tag{6.114}$$

we take the logarithm

$$\hat{H}(z) = \ln G - \ln\left(1 - \sum_{l=1}^{p} a_l z^{-l}\right) = \sum_{k=-\infty}^{\infty} \hat{h}[k] z^{-k} \tag{6.115}$$

and the derivative of both sides with respect to *z*

$$\frac{-\sum_{n=1}^{p} n a_n z^{-n-1}}{1-\sum_{l=1}^{p} a_l z^{-l}} = -\sum_{k=-\infty}^{\infty} k\hat{h}[k] z^{-k-1} \tag{6.116}$$

Multiplying both sides by $-z\left(1-\sum_{l=1}^{p} a_l z^{-l}\right)$, we obtain

$$\sum_{n=1}^{p} n a_n z^{-n} = \sum_{n=-\infty}^{\infty} n\hat{h}[n] z^{-n} - \sum_{l=1}^{p}\sum_{k=-\infty}^{\infty} k\hat{h}[k] a_l z^{-k-l} \tag{6.117}$$

which, after replacing $l = n - k$, and equating terms in z^{-1}, results in

$$\begin{aligned} n a_n &= n\hat{h}[n] - \sum_{k=1}^{n-1} k\hat{h}[k] a_{n-k} \qquad 0 < n \le p \\ 0 &= n\hat{h}[n] - \sum_{k=n-p}^{n-1} k\hat{h}[k] a_{n-k} \qquad n > p \end{aligned} \tag{6.118}$$

so that the complex cepstrum can be obtained from the LPC coefficients by the following recursion:

$$\hat{h}[n] = \begin{cases} 0 & n < 0 \\ \ln G & n = 0 \\ a_n + \sum_{k=1}^{n-1}\left(\frac{k}{n}\right)\hat{h}[k] a_{n-k} & 0 < n \le p \\ \sum_{k=n-p}^{n-1}\left(\frac{k}{n}\right)\hat{h}[k] a_{n-k} & n > p \end{cases} \tag{6.119}$$

where the value for $n = 0$ can be obtained from Eqs. (6.115) and (6.111). We note that, while there are a finite number of LPC coefficients, the number of cepstrum coefficients is infinite. Speech recognition researchers have shown empirically that a finite number is sufficient: 12 - 20 depending on the sampling rate and whether or not frequency warping is done. In Chapter 8 we discuss the use of the cepstrum in speech recognition.

This recursion should not be used in the reverse mode to compute the LPC coefficients from *any* set of cepstrum coefficients, because the recursion in Eq. (6.119) assumes an all-pole model with all poles inside the unit circle, and that might not be the case for an arbitrary cepstrum sequence, so that the recursion might yield a set of unstable LPC coefficients. In some experiments it has been shown that quantized LPC-cepstrum can yield unstable LPC coefficients over 5% of the time.

6.4.3. Cepstrum of Periodic Signals

It is important to see what the cepstrum of periodic signals looks like. To do so, let's consider the following signal:

$$x[n] = \sum_{k=0}^{M-1} \alpha_k \delta[n-kN] \tag{6.120}$$

which can be viewed as an impulse train of period *N* multiplied by an analysis window, so that only *M* impulses remain. Its *z*-transform is

$$X(z) = \sum_{k=0}^{M-1} \alpha_k z^{-kN} \tag{6.121}$$

which is a polynomial in z^{-N} rather than z^{-1}. Therefore, $X(z)$ can be expressed as a product of factors of the form $(1-a_k z^{-Nk})$ and $(1-u_k z^{Nk})$. Following the derivation in Section 6.4.2, it is clear that its complex cepstrum is nonzero only at integer multiples of *N*:

$$\hat{x}[n] = \sum_{k=-\infty}^{\infty} \beta_k \delta[n-kN] \tag{6.122}$$

A particularly interesting case is when $\alpha_k = \alpha^k$ with $0 < \alpha < 1$, so that Eq. (6.121) can be expressed as

$$X(z) = 1 + \alpha z^{-N} + \cdots + (\alpha z^{-N})^{M-1} = \frac{1-(\alpha z^{-N})^M}{1-\alpha z^{-N}} \tag{6.123}$$

so that taking the logarithm of Eq. (6.123) and expanding it in Taylor series using Eq. (6.111) results in

$$\hat{X}(z) = \ln X(z) = \sum_{r=1}^{\infty} \frac{\alpha^r}{r} z^{-rN} - \sum_{l=1}^{\infty} \frac{\alpha^{lM}}{l} z^{-lMN} = \sum_{n=1}^{\infty} \hat{x}[n] z^{-n} \tag{6.124}$$

which lets us compute the complex cepstrum as

$$\hat{x}[n] = \sum_{r=1}^{\infty} \frac{\alpha^r}{r} \delta[n-rN] - \sum_{l=1}^{\infty} \frac{\alpha^{lM}}{l} \delta[n-lMN] \tag{6.125}$$

An infinite impulse train can be obtained by making $\alpha \to 1$ and $M \to \infty$ in Eq. (6.125):

$$\hat{x}[n] = \sum_{r=1}^{\infty} \frac{\delta[n-rN]}{r} \tag{6.126}$$

We see from Eq. (6.126) that the cepstrum of an impulse train goes to 0 as *n* increases. This justifies our assumption of homomorphic filtering.

6.4.4. Cepstrum of Speech Signals

We can compute the cepstrum of a speech segment by windowing the signal with a window of length *N*. In practice, the cepstrum is not computed through Eq. (6.112), since root-finding algorithms are slow and offer numerical imprecision for the large values of *N* used. Instead, we can compute the cepstrum directly through its definition of Eq. (6.105), using the DFT as follows:

$$X_a[k] = \sum_{n=0}^{N-1} x[n] e^{-j2\pi nk/N} \;, \qquad 0 \le k < N \tag{6.127}$$

$$\hat{X}_a[k] = \ln X_a[k] \,, \qquad 0 \le k < N \tag{6.128}$$

$$\hat{x}_a[n] = \frac{1}{N} \sum_{n=0}^{N-1} \hat{X}_a[k] e^{-j2\pi nk/N} \;, \qquad 0 \le n < N \tag{6.129}$$

The subscript a means that the new complex cepstrum $\hat{x}_a[n]$ is an aliased version of $\hat{x}[n]$ given by

$$\hat{x}_a[n] = \sum_{r=-\infty}^{\infty} \hat{x}[n+rN] \tag{6.130}$$

which can be derived by using the sampling theorem of Chapter 5, by reversing the concepts of time and frequency.

This aliasing introduces errors in the estimation that can be reduced by choosing a large value for *N*.

Computation of the complex cepstrum requires computing the complex logarithm and, in turn, the phase. However, given the principal value of the phase $\theta_p[k]$, there are infinite possible values for $\theta[k]$:

$$\theta[k] = \theta_p[k] + 2\pi n_k \tag{6.131}$$

From Chapter 5 we know that if $x[n]$ is real, $\arg\left[X(e^{j\omega}) \right]$ is an odd function and also continuous. Thus we can do *phase unwrapping* by choosing n_k to guarantee that $\theta[k]$ is a smooth function, i.e., by forcing the difference between adjacent values to be small:

$$\left|\theta[k] - \theta[k-1]\right| < \pi \tag{6.132}$$

A linear phase term *r* as in Eq. (6.110), would contribute to the phase difference in Eq. (6.132) with $2\pi r / N$, which may result in errors in the phase unwrapping if $\theta[k]$ is changing sufficiently rapidly. In addition, there could be large changes in the phase difference if $X_a[k]$ is noisy. To guarantee that we can track small phase differences, a value of *N* several times larger than the window size is required: *i.e.*, the input signal has to be zero-padded prior to the FFT computation. Finally, the delay r in Eq. (6.109), can be obtained by forcing the phase to be an odd function, so that:

$$\theta[N/2] = \pi r \tag{6.133}$$

For unvoiced speech, the unwrapped phase is random, and therefore only the real cepstrum has meaning. In practical situations, even voiced speech has some frequencies at which noise dominates (typically very low and high frequencies), which results in phase $\theta[k]$ that changes drastically from frame to frame. Because of this, the complex cepstrum in Eq. (6.105) is rarely used for real speech signals. Instead, the real cepstrum is used much more often:

$$C_a[k] = \ln\left|X_a[k]\right|, \qquad 0 \le k < N \tag{6.134}$$

$$c_a[n] = \frac{1}{N}\sum_{n=0}^{N-1} C_a[k] e^{-j2\pi nk/N}, \qquad 0 \le n < N \tag{6.135}$$

Similarly, it can be shown that for the new real cepstrum $c_a[n]$ is an aliased version of $c[n]$ given by

$$c_a[n] = \sum_{r=-\infty}^{\infty} c[n+rN] \tag{6.136}$$

which again has aliasing that can be reduced by choosing a large value for *N*.

6.4.5. Source-Filter Separation via the Cepstrum

We have seen that, if the filter is a rational transfer function, and the source is an impulse train, the homomorphic filtering of Figure 6.24 can approximately separate them. Because of problems in estimating the phase in speech signals (see Section 6.4.4), we generally compute the real cepstrum using Eqs. (6.127), (6.134) and (6.135), and then compute the complex cepstrum under the assumption of a minimum phase signal according to Eq. (6.113). The result of separating source and filter using this cepstral deconvolution is shown in Figure 6.25 for voiced speech and Figure 6.26 for unvoiced speech.

The real cepstrum of white noise $x[n]$ with an expected magnitude spectrum $|X(e^{j\omega})| = 1$ is 0. If colored noise is present, the cepstrum of the observed colored noise $\hat{y}[n]$ is identical to the cepstrum of the coloring filter $\hat{h}[n]$, except for a gain factor. The above is correct if we take an infinite number of noise samples, but in practice, this cannot be done and a limited number have to be used, so that this is only an approximation, though it is often used in speech processing algorithms.

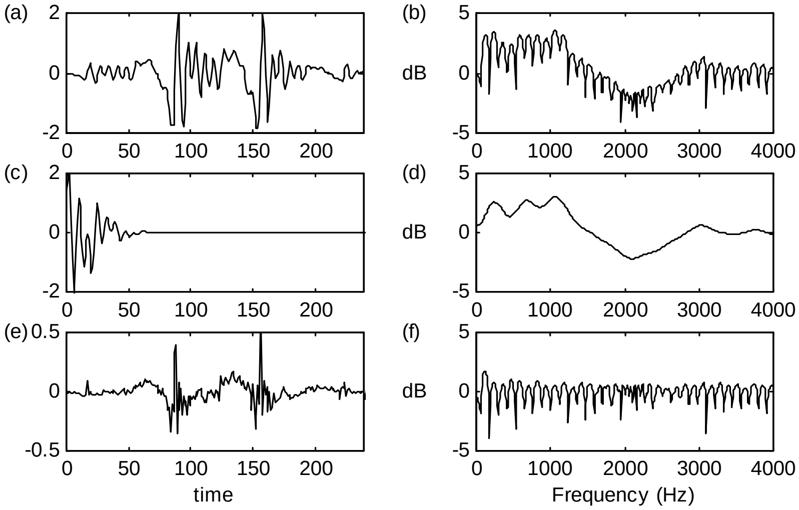

Figure 6.25 Separation of source and filter using homomorphic filtering for voiced speech with the scheme of Figure 6.24 with $N = 20$ in the homomorphic filter of Eq. (6.102) with the real cepstrum: (a) windowed signal, (b) log-spectrum, (c) filter's impulse response, (d) smoothed log-spectrum, (e) windowed excitation signal, (f) log-spectrum of high-part of cepstrum. Note that the windowed excitation is not a windowed impulse train because of the minimum phase assumption.

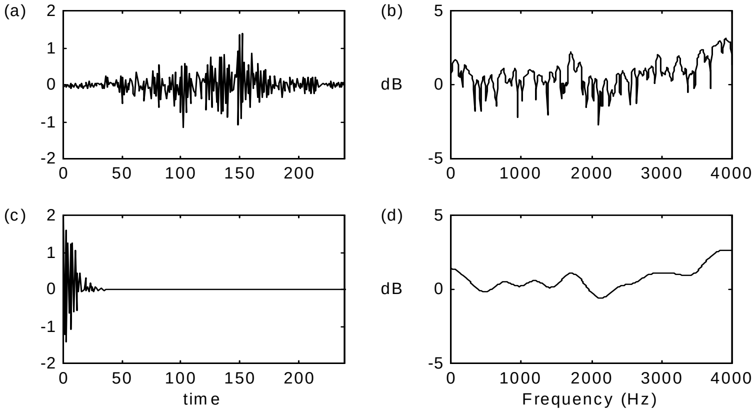

Figure 6.26 Separation of source and filter using homomorphic filtering for unvoiced speech with the scheme of Figure 6.24 with $N = 20$ in the homomorphic filter of Eq. (6.102) with the real cepstrum: (a) windowed signal, (b) log-spectrum, (c) filter's impulse response, (d) smoothed log-spectrum.

6.5. Perceptually-Motivated Representations

In this section we describe some aspects of human perception, and methods motivated by the behavior of the human auditory system: Mel-Frequency Cepstrum Coefficients (MFCC) and Perceptual Linear Prediction (PLP). These methods have been successfully used in speech recognition. First we present several nonlinear frequency scales that have been used in such representations.

6.5.1. The Bilinear Transform

The transformation

$$s = \frac{z^{-1} - \alpha}{1 - \alpha z^{-1}} \tag{6.137}$$

for $0 < \alpha < 1$ belongs to the class of *bilinear* transforms. It is a mapping in the complex plane that maps the unit circle onto itself. The frequency transformation is obtained by making the substitution $z = e^{j\omega}$ and $s = e^{j\Omega}$:

$$\Omega = \omega + 2\arctan\left[\frac{\alpha\sin(\omega)}{1 - \alpha\cos(\omega)}\right] \tag{6.138}$$

This transformation is very similar to the Bark and mel scale for an appropriate choice of the parameter α (see Chapter 2). Oppenheim [31] showed that the advantage of this transformation is that it can be used to transform a time sequence in the linear frequency into another time sequence in the warped frequency, as shown in Figure 6.27. This bilinear transform has been successfully applied to cepstral and autocorrelation coefficients.

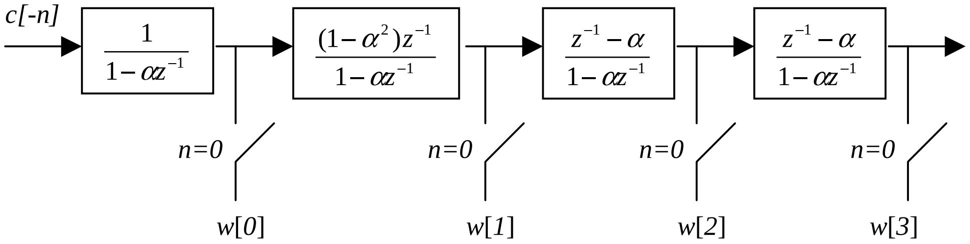
c[-n]
$\frac{1}{1-\alpha z^{-1}}$
$\frac{(1-\alpha^2)z^{-1}}{1-\alpha z^{-1}}$
$\frac{z^{-1}-\alpha}{1-\alpha z^{-1}}$
$\frac{z^{-1}-\alpha}{1-\alpha z^{-1}}$
n=0
n=0
n=0
n=0
w[0]
w[1]
w[2]
w[3]

Figure 6.27 Implementation of the frequency-warped cepstral coefficients as a function of the linear-frequency cepstrum coefficients. Both sets of coefficients are causal. The input is the time-reversed cepstrum sequence, and the output can be obtained by sampling the outputs of the filters at time $n = 0$. The filters used for $w[m]$ $m > 2$ are the same. Note that, for a finite-length cepstrum, an infinite-length warped cepstrum results.

For a finite number of cepstral coefficients the bilinear transform in Figure 6.27 results in an infinite number of warped cepstral coefficients. Since truncation is usually done in practice, the bilinear transform is equivalent to a matrix multiplication, where the matrix is a function of the warping parameter α. Shikano [43] showed these warped cepstral coefficients were beneficial for speech recognition.

6.5.2. Mel-Frequency Cepstrum

The *Mel-Frequency Cepstrum Coefficients* (MFCC) is a representation defined as the real cepstrum of a windowed short-time signal derived from the FFT of that signal. The difference from the real cepstrum is that a nonlinear frequency scale is used, which approximates the behavior of the auditory system. Davis and Mermelstein [8] showed the MFCC representation to be beneficial for speech recognition.

Given the DFT of the input signal

$$X_a[k] = \sum_{n=0}^{N-1} x[n] e^{-j2\pi nk/N}, \qquad 0 \le k < N \tag{6.139}$$

we define a filterbank with M filters ($m = 1, 2, \cdots, M$), where filter m is triangular filter given by:

$$H_m[k] = \begin{cases} 0 & k < f[m-1] \\ \dfrac{2\left(k - f[m-1]\right)}{\left(f[m+1] - f[m-1]\right)\left(f[m] - f[m-1]\right)} & f[m-1] \le k \le f[m] \\ \dfrac{2\left(f[m+1] - k\right)}{\left(f[m+1] - f[m-1]\right)\left(f[m+1] - f[m]\right)} & f[m] \le k \le f[m+1] \\ 0 & k > f[m+1] \end{cases} \tag{6.140}$$

Such filters compute the average spectrum around each center frequency with increasing bandwidths, and they are displayed in Figure 6.28.

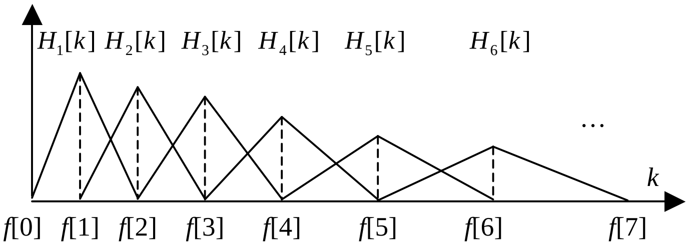

Figure 6.28 Triangular filters used in the computation of the mel-cepstrum using Eq. (6.140).

Alternatively, the filters can be chosen as

$$H_m^{'}[k] = \begin{cases} 0 & k < f[m-1] \\ \dfrac{(k - f[m-1])}{(f[m] - f[m-1])} & f[m-1] \le k \le f[m] \\ \dfrac{(f[m+1] - k)}{(f[m+1] - f[m])} & f[m] \le k \le f[m+1] \\ 0 & k > f[m+1] \end{cases} \tag{6.141}$$

which satisfies $\sum_{m=0}^{M-1} H_m^{'}[k] = 1$. The mel-cepstrum computed with $H_m[k]$ or $H_m^{'}[k]$ will differ by a constant vector for all inputs, so the choice becomes unimportant when used in a speech recognition system that has trained with the same filters.

Let's define f_l and f_h to be the lowest and highest frequencies of the filterbank in Hz, F_s the sampling frequency in Hz, *M* the number of filters, and *N* the size of the FFT. The boundary points *f*[*m*] are uniformly spaced in the mel-scale:

$$f[m] = \left(\frac{N}{F_s}\right) B^{-1}\left(B(f_l) + m\frac{B(f_h) - B(f_l)}{M+1} \right) \tag{6.142}$$

where the mel-scale *B* is given by Eq. (2.6), and B^{-1} is its inverse

$$B^{-1}(b) = 700\left(\exp(b/1125) - 1\right) \tag{6.143}$$

We then compute the log-energy at the output of each filter as

$$S[m] = \ln\left[\sum_{k=0}^{N-1} \left| X_a[k] \right|^2 H_m[k] \right], \qquad 0 \le m < M \tag{6.144}$$

The mel frequency cepstrum is then the discrete cosine transform of the *M* filter outputs:

$$c[n] = \sum_{m=0}^{M-1} S[m] \cos\left(\pi n(m+1/2)/M\right) \qquad 0 \le n < M \tag{6.145}$$

where *M* varies for different implementations from 24 to 40. For speech recognition, typically only the first 13 cepstrum coefficients are used. It is important to note that the MFCC representation is no longer a homomorphic transformation. It would be if the order of summation and logarithms in Eq. (6.144) were reversed:

$$S[m] = \sum_{k=0}^{N-1} \ln\left(\left| X_a[k] \right|^2 H_m[k] \right) \qquad 0 \le m < M \tag{6.146}$$

In practice, however, the MFCC representation is approximately homomorphic for filters that have a smooth transfer function. The advantage of the MFCC representation using

(6.144) instead of (6.146) is that the filter energies are more robust to noise and spectral estimation errors. This algorithm has been used extensively as a feature vector for speech recognition systems.

While the definition of cepstrum in Section 6.4.1 uses an inverse DFT, since $S[m]$ is even, a DCT-II can be used instead (see Chapter 5).

6.5.3. Perceptual Linear Prediction (PLP)

Perceptual Linear Prediction (PLP) [16] uses the standard Durbin recursion of Section 6.3.2.2 to compute LPC coefficients, and typically the LPC coefficients are transformed to LPC-cepstrum using the recursion in Section 6.4.2.1. But unlike standard linear prediction, the autocorrelation coefficients are not computed in the time domain through Eq. (6.55).

The autocorrelation $R_x[n]$ is the inverse Fourier transform of the power spectrum $\left|X(\omega)\right|^2$ of the signal. We cannot compute the continuous-frequency Fourier transform easily, but we can take an FFT to compute $X[k]$, so that the autocorrelation can be obtained as the inverse Fourier transform of $\left|X[k]\right|^2$. Since the discrete Fourier transform is not performing linear convolution but circular convolution, we need to make sure that the FFT size is larger than twice the window length (see Section 5.3.4) for this to hold. This alternate way of computing autocorrelation coefficients, entailing two FFTs and N multiplies and adds, should yield identical results. Since normally only a small number p of autocorrelation coefficients are needed, this is generally not a cost-effective way to do it, unless the first FFT has to be computed for other reasons.

Perceptual linear prediction uses the above method, but replaces $\left|X[k]\right|^2$ by a perceptually motivated power spectrum. The most important aspect is the non-linear frequency scaling, which can be achieved through a set of filterbanks similar to those described in Section 6.5.2, so that this critical-band power spectrum can be sampled in approximately 1-bark intervals. Another difference is that, instead of taking the logarithm on the filterbank energy outputs, a different non-linearity compression is used, often the cubic root. It is reported [16] that the use of this different non-linearity is beneficial for speech recognizers in noisy conditions.

6.6. FORMANT FREQUENCIES

Formant frequencies are the resonances in the vocal tract and, as we saw in Chapter 2, they convey the differences between different sounds. Expert spectrogram readers are able to recognize speech by looking at a spectrogram, particularly at the formants. It has been argued that they are very useful features for speech recognition, but they haven't been widely used because of the difficulty in estimating them.

One way of obtaining formant candidates at a frame level is to compute the roots of a p^{th}-order LPC polynomial [3, 26]. There are standard algorithms to compute the complex roots of a polynomial with real coefficients [36], though convergence is not guaranteed. Each complex root z_i can be represented as

$$z_i = \exp(-\pi b_i + j2\pi f_i) \tag{6.147}$$

where f_i and b_i are the formant frequency and bandwidth, respectively, of the i^{th} root. Real roots are discarded and complex roots are sorted by increasing *f*, discarding negative values. The remaining pairs (f_i, b_i) are the formant candidates. Traditional formant trackers discard roots whose bandwidths are higher than a threshold [46], say 200 Hz.

Closed-phase analysis of voiced speech [5] uses only the regions for which the glottis is closed and thus there is no excitation. When the glottis is open, there is a coupling of the vocal tract with the lungs and the resonance bandwidths are somewhat larger. Determination of the closed-phase regions directly from the speech signal is difficult, so often an *electroglottograph* (EGG) signal is used [23]. EGG signals, obtained by placing electrodes at the speaker's throat, are very accurate in determining the times when the glottis is closed. Using samples in the closed-phase covariance analysis can yield accurate results [46]. For female speech, the closed-phase is short, and sometimes non-existent, so such analysis can be a challenge. EGG signals are useful also for pitch tracking and are described in more detail in Chapter 16.

Another common method consists of finding the peaks on a smoothed spectrum, such as that obtained through an LPC analysis [26, 40]. The advantage of this method is that you can always compute the peaks and it is more computationally efficient than extracting the complex roots of a polynomial. On the other hand, this procedure generally doesn't estimate the formant's bandwidth. The first three formants are typically estimated this way for formant synthesis (see Chapter 16), since they are the ones that allow sound classification, whereas the higher formants are more speaker dependent.

Sometimes, the signal goes through some *conditioning*, which includes sampling rate conversion to remove frequencies outside the range we are interested in. For example, if we are interested only in the first three formants, we can safely downsample the input signal to 8 kHz, since we know all three formants should be below 4 kHz. This downsampling reduces computation and the chances of the algorithm to find formant values outside the expected range (otherwise peaks or roots could be chosen above 4 kHz which we know do not correspond to any of the first three formants). Pre-emphasis filtering is also often used to whiten the signal.

Because of the thresholds imposed above, it is possible that the formants are not continuous. For example, when the vocal tract's spectral envelope is changing rapidly, bandwidths obtained through the above methods are overestimates of the true bandwidths, and they may exceed the threshold and thus be rejected. It is also possible for the peak-picking algorithm to classify a harmonic as a formant during some regions where it is much stronger than the other harmonics. Due to the thresholds used, a given frame could have no formants, only one formant (either first, second, or third), two, three, or more. Formant alignment from one frame to another has often been done using heuristics to prevent such discontinuities.

6.6.1. Statistical Formant Tracking

It is desirable to have an approach that does not use any thresholds on formant candidates and uses a probabilistic model to do the tracking instead of heuristics [1]. The formant candidates can be obtained from roots of the LPC polynomial, peaks in the smoothed spectrum or even from a dense sample of possible points. If the first *n* formants are desired, and we have *(p/2)* formant candidates, a maximum of *r n*-tuples are considered, where *r* is given by

$$r = \begin{pmatrix} p/2 \\ n \end{pmatrix} \tag{6.148}$$

A Viterbi search (see Chapter 8) is then carried out to find the most likely path of formant *n*-tuples given a model with some a priori knowledge of formants. The prior distribution for formant targets is used to determine which formant candidate to use of all possible choices for the given phoneme (*i.e.*, we know that F1 for an AE should be around 800 Hz). Formant continuity is imposed through the prior distribution of the formant slopes. This algorithm produces *n* formants for every frame, including silence.

Since we are interested in obtaining the first three formants (*n*=3) and F3 is known to be lower than 4 kHz, it is advantageous to downsample the signal to 8 kHz in order to avoid obtaining formant candidates above 4 kHz and to let us use a lower-order analysis which offers fewer numerical problems when computing the roots. With *p* = 14, it results in a maximum of *r* = 35 triplets for the case of no real roots.

Let **X** be a sequence of *T* feature vectors $\mathbf{x}_t$ of dimension *n*:

$$\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T)' \tag{6.149}$$

where the prime denotes transpose.

We estimate the formants with the knowledge of what sound occurs at that particular time, for example by using a speech recognizer that segments the waveform into different phonemes (see Chapter 9) or states q_t within a phoneme. In this case we assume that the output distribution of each state *i* is modeled by one Gaussian density function with a mean μ_i and covariance matrix Σ_i. We can define up to *N* states, with λ being the set of all means and covariance matrices for all:

$$\lambda = (\mu_1, \Sigma_1, \mu_2, \Sigma_2, \cdots, \mu_N, \Sigma_N) \tag{6.150}$$

Therefore, the log-likelihood for **X** is given by

$$\ln p(\mathbf{X} \mid \hat{\mathbf{q}}, \lambda) = -\frac{TM}{2}\ln(2\pi) - \frac{1}{2}\sum_{t=1}^{T} \ln |\Sigma_{q_t}| - \frac{1}{2}\sum_{t=1}^{T} (\mathbf{x}_t - \mu_{q_t})' \Sigma_{q_t}^{-1} (\mathbf{x}_t - \mu_{q_t}) \tag{6.151}$$

Maximizing **X** in Eq. (6.151) leads to the trivial solution $\hat{\mathbf{X}} = (\mu_{q_1}, \mu_{q_2}, \ldots, \mu_{q_T})'$, a piecewise function whose value is that of the best *n*-tuple candidate. This function has discontinuities at state boundaries and thus is not likely to represent well the physical phenomena of speech.

This problem arises because the slopes at state boundaries do not match the slopes of natural speech. To avoid these discontinuities, we would like to match not only the target formants at each state, but also the formant slopes at each state. To do that, we augment the feature vector $\mathbf{x}_t$ at frame t with the delta vector $\mathbf{x}_t - \mathbf{x}_{t-1}$. Thus, we increase the parameter space of λ with the corresponding means δ_i and covariance matrices Γ_i of these delta parameters, and assume statistical independence among them. The corresponding new log-likelihood has the form

$$\begin{aligned}\ln p(\mathbf{X} \mid \hat{\mathbf{q}}, \lambda) = K - \frac{1}{2}\sum_{t=1}^{T} \ln |\Sigma_{q_t}| - \frac{1}{2}\sum_{t=2}^{T} \ln |\Gamma_{q_t}| \\ - \frac{1}{2}\sum_{t=1}^{T} (\mathbf{x}_t - \mu_{q_t})' \Sigma_{q_t}^{-1} (\mathbf{x}_t - \mu_{q_t}) - \frac{1}{2}\sum_{t=2}^{T} (\mathbf{x}_t - \mathbf{x}_{t-1} - \delta_{q_t})' \Gamma_{q_t}^{-1} (\mathbf{x}_t - \mathbf{x}_{t-1} - \delta_{q_t})\end{aligned} \tag{6.152}$$

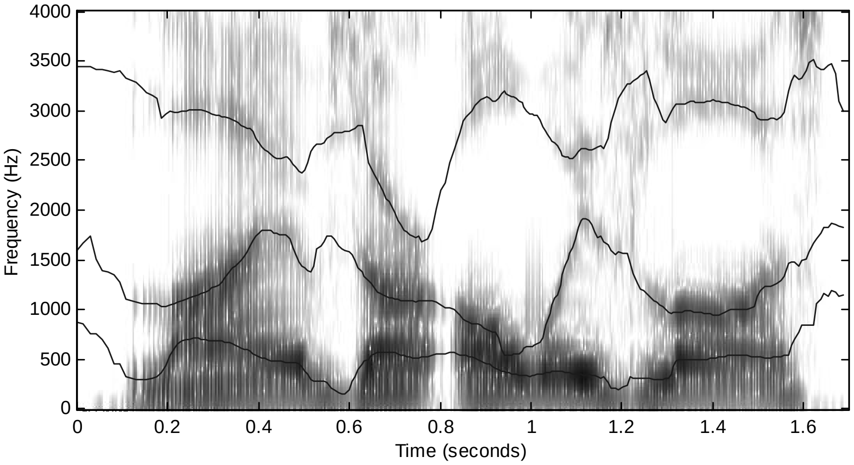

Figure 6.29 Spectrogram and three smoothed formants.

Maximization of Eq. (6.152) with respect to $\mathbf{x}_t$ requires solving several sets of linear equations. If Γ_i and Σ_i are diagonal covariance matrices, it results in a set of linear equations for each of the M dimensions

$$\mathbf{BX} = \mathbf{c} \tag{6.153}$$

where **B** is a tridiagonal matrix (all values are zero except for those in the main diagonal and its two adjacent diagonals), which leads to a very efficient solution [36]. For example, the values of **B** and **c** for $T = 3$ are given by

$$\mathbf{B} = \begin{pmatrix} \frac{1}{\sigma_{q_1}^2} + \frac{1}{\gamma_{q_2}^2} & -\frac{1}{\gamma_{q_2}^2} & 0 \\ -\frac{1}{\gamma_{q_2}^2} & \frac{1}{\sigma_{q_2}^2} + \frac{1}{\gamma_{q_2}^2} + \frac{1}{\gamma_{q_3}^2} & -\frac{1}{\gamma_{q_3}^2} \\ 0 & -\frac{1}{\gamma_{q_3}^2} & \frac{1}{\sigma_{q_3}^2} + \frac{1}{\gamma_{q_3}^2} \end{pmatrix} \tag{6.154}$$

$$\mathbf{c} = \begin{pmatrix} \frac{\mu_{q_1}}{\sigma_{q_1}^2} - \frac{\delta_{q_2}}{\gamma_{q_2}^2} & \frac{\mu_{q_2}}{\sigma_{q_2}^2} + \frac{\delta_{q_2}}{\gamma_{q_2}^2} - \frac{\delta_{q_3}}{\gamma_{q_3}^2} & \frac{\mu_{q_3}}{\sigma_{q_3}^2} + \frac{\delta_{q_3}}{\gamma_{q_3}^2} \end{pmatrix}' \tag{6.155}$$

where just one dimension is represented, and the process is repeated for all dimensions with a computational complexity of $O(TM)$.

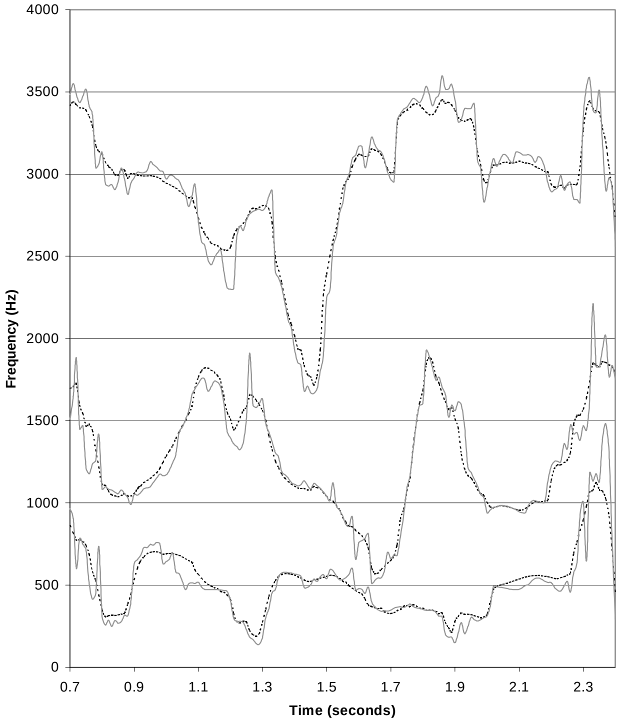

Figure 6.30 Raw formants (ragged gray line) and smoothed formants (dashed line).

The maximum likelihood sequence $\hat{\mathbf{x}}_t$ is close to the targets μ_i while keeping the slopes close to δ_i for a given state i, thus estimating a continuous function. Because of the delta coefficients, the solution depends on all the parameters of all states and not just the current state. This procedure can be performed for the formants as well as the bandwidths.

The parameters μ_i, Σ_i, δ_i, and Γ_i can be re-estimated using the EM algorithm described in Chapter 8. In [1] it is reported that two or three iterations are sufficient for speaker-dependent data.

The formant track obtained through this method can be rough, and it may be desired to smooth it. Smoothing without knowledge about the speech signal would result in either blurring the sharp transitions that occur in natural speech, or maintaining ragged formant tracks where the underlying physical phenomena vary slowly with time. Ideally we would like a larger adjustment to the raw formant when the error in the estimate is large relative to the variance of the corresponding state within a phoneme. This can be done by modeling the formant measurement error as a Gaussian distribution. Figure 6.29 shows an utterance from a male speaker with the smoothed formant tracks, and Figure 6.30 compares the raw and smoothed formants. When no real formant is visible from the spectrogram, the algorithm tends to assign a large bandwidth (not shown in the figure).

6.7. THE ROLE OF PITCH

Pitch determination is very important for many speech processing algorithms. The concatenative speech synthesis methods of Chapter 16 require pitch tracking on the desired speech segments if prosody modification is to be done. Chinese speech recognition systems use pitch tracking for tone recognition, which is important in disambiguating the myriad of homophones. Pitch is also crucial for prosodic variation in text-to-speech systems (see Chapter 15) and spoken language systems (see Chapter 17). While in the previous sections we have dealt with features representing the filter, pitch represents the source of the model illustrated in Figure 6.1.

Pitch determination algorithms also use short-term analysis techniques, which means that for every frame $\mathbf{x}_m$ we get a score $f(T \mid \mathbf{x}_m)$ that is a function of the candidate pitch periods *T*. These algorithms determine the optimal pitch by maximizing

$$T_m = \arg\max_T f(T \mid \mathbf{x}_m) \tag{6.156}$$

We describe several such functions computed through the autocorrelation method and the normalized cross-correlation method, as well as the signal conditioning that is often performed. Other approaches based on cepstrum [28] have also been used successfully. a good summary of techniques used for pitch tracking is provided by [17, 45].

Pitch determination using Eq. (6.156) is error prone, and a smoothing stage is often done. This smoothing, described in Section 6.7.4, takes into consideration that the pitch does not change quickly over time.

6.7.1. Autocorrelation Method

A commonly used method to estimate pitch is based on detecting the highest value of the autocorrelation function in the region of interest. This region must exclude $m = 0$, as that is the absolute maximum of the autocorrelation function [37]. As discussed in Chapter 5, the statistical autocorrelation of a sinusoidal random process

$$\mathbf{x}[n] = \cos(\omega_0 n + \varphi) \tag{6.157}$$

is given by

$$R[m] = E\{\mathbf{x}^*[n]\mathbf{x}[n+m]\} = \frac{1}{2}\cos(\omega_0 m) \tag{6.158}$$

which has maxima for $m = lT_0$, the pitch period and its harmonics, so that we can find the pitch period by computing the highest value of the autocorrelation. Similarly, it can be shown that any WSS periodic process $\mathbf{x}[n]$ with period T_0 also has an autocorrelation $R[m]$ which exhibits its maxima at $m = lT_0$.

In practice, we need to obtain an estimate $\hat{R}[m]$ from knowledge of only N samples. If we use a window $w[n]$ of length N on $\mathbf{x}[n]$ and assume it to be real, the empirical autocorrelation function is given by

$$\hat{R}[m] = \frac{1}{N}\sum_{n=0}^{N-1-|m|} w[n]\mathbf{x}[n]w[n+|m|]\mathbf{x}[n+|m|] \tag{6.159}$$

whose expected value can be shown to be

$$E\left\{\hat{R}[m]\right\} = R[m]\left(w[m] * w[-m]\right) \tag{6.160}$$

where

$$w[m] * w[-m] = \sum_{n=0}^{N-|m|-1} w[n]w[n+|m|] \tag{6.161}$$

which, for the case of a rectangular window of length N, is given by

$$w[m] * w[-m] = \begin{cases} 1 - \frac{|m|}{N} & |m| < N \\ 0 & |m| \geq N \end{cases} \tag{6.162}$$

which means that $\hat{R}[m]$ is a biased estimator of $R[m]$. So, if we compute the peaks based on Eq. (6.159), the estimate of the pitch will also be biased. Although the variance of the estimate is difficult to compute, it is easy to see that as m approaches N, fewer and fewer samples of $x[n]$ are involved in the calculation, and thus the variance of the estimate is expected to increase. If we multiply Eq. (6.159) by $N/(N-m)$, the estimate will be unbiased but the variance will be larger.

Using the empirical autocorrelation in Eq. (6.159) for the random process in Eq. (6.157) results in an expected value of

$$E\left\{\hat{R}[m]\right\} = \left(1 - \frac{|m|}{N}\right)\frac{\cos(\omega_0 m)}{2}, \qquad |m| < N \tag{6.163}$$

whose maximum coincides with the pitch period for $m > m_0$.

Since pitch periods can be as low as 40 Hz (for a very low-pitched male voice) or as high as 600 Hz (for a very high-pitched female or child's voice), the search for the maximum is conducted within a region. This F0 detection algorithm is illustrated in Figure 6.31 where the lag with highest autocorrelation is plotted for every frame. In order to see periodicity present in the autocorrelation, we need to use a window that contains at least two pitch periods, which, if we want to detect a 40Hz pitch, implies 50ms (see Figure 6.32). For window lengths so long, the assumption of stationarity starts to fail, because a pitch period at the beginning of the window can be significantly different than at the end of the window. One possible solution to this problem is to estimate the autocorrelation function with different window lengths for different lags *m*.

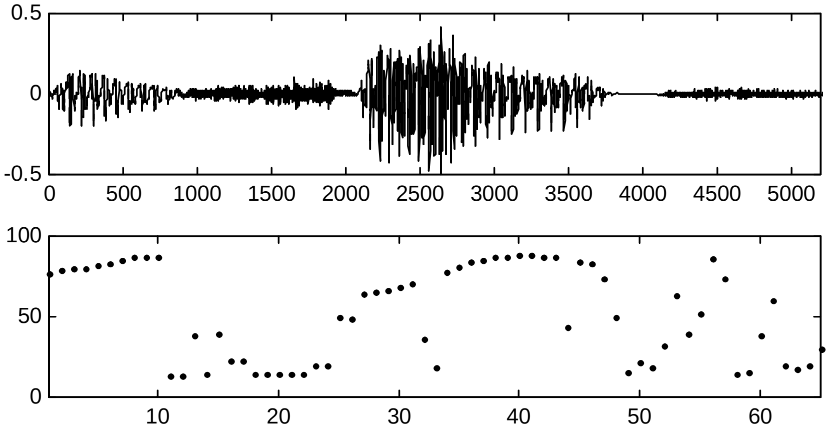

Figure 6.31 Waveform and unsmoothed pitch track with the autocorrelation method. A frame shift of 10 ms, a Hamming window of 30 ms, and a sampling rate of 8kHz were used. Notice that two frames in the voiced region have an incorrect pitch. The pitch values in the unvoiced regions are essentially random.

The candidate pitch periods in Eq. (6.156) can be simply $T_m = m$; i.e., the pitch period is any integer number of samples. For low values of T_m, the frequency resolution is lower than for high values. To maintain a relatively constant frequency resolution, we do not have to search all the pitch periods for large T_m. Alternatively, if the sampling frequency is not high, we may need to use fractional pitch periods (often done in the speech coding algorithms of Chapter 7)

The autocorrelation function can be efficiently computed by taking a signal, windowing it, and taking an FFT and then the square of the magnitude.

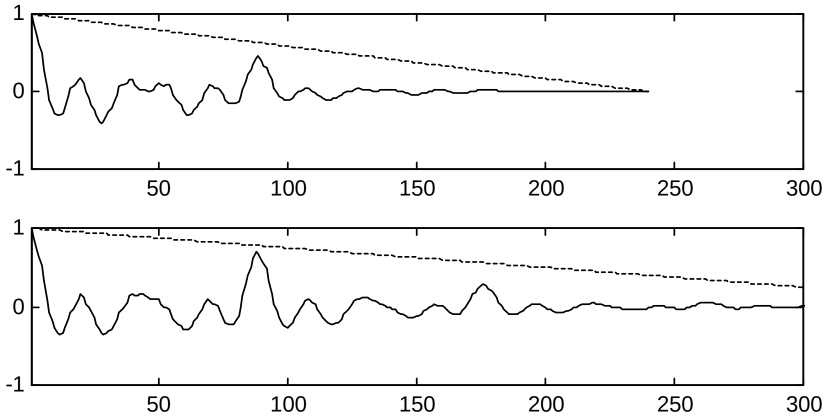

Figure 6.32 Autocorrelation function for frame 40 in Figure 6.31. The maximum occurs at 89 samples. A sampling frequency of 8 kHz and window shift of 10ms are used. The top figure is using a window length of 30 ms, whereas the bottom one is using 50 ms. Notice the quasi-periodicity in the autocorrelation function.

6.7.2. Normalized Cross-Correlation Method

A method that is free from these border problems and has been gaining in popularity is based on the *normalized cross-correlation* [2]

$$\alpha_t(T) = \cos(\theta) = \frac{< \mathbf{x}_t, \mathbf{x}_{t-T} >}{|\mathbf{x}_t||\mathbf{x}_{t-T}|} \tag{6.164}$$

where $\mathbf{x}_t = \{x[t-N/2], x[t-N/2+1], \cdots, x[t+N/2-1]\}$ is a vector of N samples centered at time t, and $< \mathbf{x}_t, \mathbf{x}_{t-T} >$ is the inner product between the two vectors defined as

$$< \mathbf{x}_n, \mathbf{y}_l > \sum_{m=-N/2}^{N/2-1} x[n+m]y[l+m] \tag{6.165}$$

so that, using Eq. (6.165), the normalized cross-correlation can be expressed as

$$\alpha_t(T) = \frac{\sum_{n=-N/2}^{N/2-1} x[t+n]x[t+n-T]}{\sqrt{\sum_{n=-N/2}^{N/2-1} x^2[t+n] \sum_{m=-N/2}^{N/2-1} x^2[t+m+T]}} \tag{6.166}$$

where we see that the numerator in Eq. (6.166) is very similar to the autocorrelation in Section 6.7.1, but where *N* terms are used in the addition for all values of *T*.

The maximum of the normalized cross-correlation method is shown in Figure 6.33 (b). Unlike the autocorrelation method, the estimate of the normalized cross-correlation is not biased by the term $(1-m/N)$. For perfectly periodic signals, this results in identical values of the normalized cross-correlation function for *kT*. This can result in pitch halving, where 2*T* can be chosen as the pitch period, which happens in Figure 6.33 (b) at the beginning of the utterance. Using a decaying bias $(1-m/M)$ with $M \gg N$, can be useful in reducing pitch halving, as we see in Figure 6.33 (c).

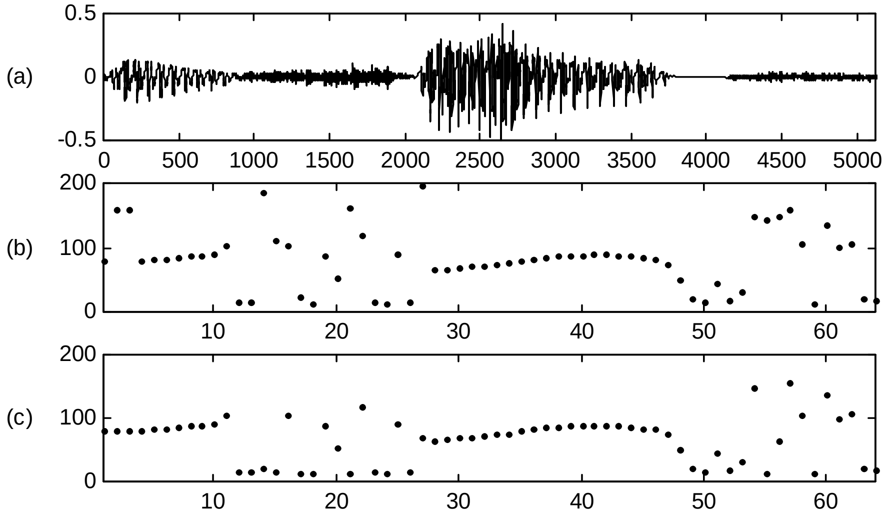

Figure 6.33 (a) Waveform and (b) (c) unsmoothed pitch tracks with the normalized cross-correlation method. A frame shift of 10 ms, window length of 10 ms, and sampling rate of 8 kHz were used. (b) is the standard normalized cross-correlation method, whereas (c) has a decaying term. If we compare it to the autocorrelation method of Figure 6.31, the middle voiced region is correctly identified in both (b) and (c), but two frames at the beginning of (b) that have pitch halving are eliminated with the decaying term. Again, the pitch values in the unvoiced regions are essentially random.

Because the number of samples involved in the calculation is constant, this estimate is unbiased and has lower variance than that of the autocorrelation. Unlike the autocorrelation method, the window length could be lower than the pitch period, so that the assumption of stationarity is more accurate and it has more time resolution. While pitch trackers based on the normalized cross-correlation typically perform better than those based on the autocorrelation, they also require more computation, since all the autocorrelation lags can be efficiently computed through 2 FFTs and *N* multiplies and adds (see Section 5.3.4).

Let's gain some insight about the normalized cross-correlation. If $x[n]$ is periodic with period *T*, then we can predict it from a vector *T* samples in the past as:

$$\mathbf{x}_t = \rho \mathbf{x}_{t-T} + \mathbf{e}_t \tag{6.167}$$

where ρ is the prediction gain. The normalized cross-correlation measures the angle between the two vectors, as can be seen in Figure 6.34, and since it is a cosine, it has the property that $-1 \le \alpha_n(P) \le 1$.

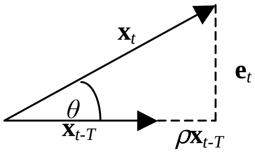

Figure 6.34 The prediction of $\mathbf{x}_t$ with $\mathbf{x}_{t-T}$ results in an error $\mathbf{e}_t$.

If we choose the value of the prediction gain ρ so as to minimize the prediction error

$$\left|\mathbf{e}_t\right|^2 = \left|\mathbf{x}_t\right|^2 - \left|\mathbf{x}_t\right|^2 \cos^2(\theta) = \left|\mathbf{x}_t\right|^2 - \left|\mathbf{x}_t\right|^2 \alpha_t^2(T) \tag{6.168}$$

and assume $\mathbf{e}_t$ is a zero-mean Gaussian random vector with a standard deviation $\sigma \,|\, \mathbf{x}_t \,|$, then

$$\ln f(\mathbf{x}_t \,|\, T) = K + \frac{\alpha_t^2(T)}{2\sigma^2} \tag{6.169}$$

so that the maximum likelihood estimate corresponds to finding the value T with highest normalized cross-correlation. Using Eq. (6.166), it is possible that $\alpha_t(T) < 0$. In this case, there is negative correlation between $\mathbf{x}_t$ and $\mathbf{x}_{t-T}$, and it is unlikely that T is a good choice for pitch. Thus, we need to force $\rho > 0$, so that Eq. (6.169) is converted into

$$\ln f(\mathbf{x}_t \,|\, T) = K + \frac{\left(\max(0, \alpha_t(T))\right)^2}{2\sigma^2} \tag{6.170}$$

The normalized cross-correlation of Eq. (6.164) predicts the current frame with a frame that occurs T samples before. Voiced speech may exhibit low correlation with a previous frame at a spectral discontinuity, such as those appearing at stops. To account for this, an enhancement can be done to consider not only the *backward* normalized cross-correlation, but also the *forward* normalized cross-correlation, by looking at a frame that occurs T samples ahead of the current frame, and taking the highest of both.

$$\ln f(\mathbf{x}_t \,|\, T) = K + \frac{\left(\max(0, \alpha_t(T), \alpha_t(-T))\right)^2}{2\sigma^2} \tag{6.171}$$

6.7.3. Signal Conditioning

Noise in the signal tends to make pitch estimation less accurate. To reduce this effect, signal conditioning or pre-processing has been proposed prior to pitch estimation [44]. Typically this involves bandpass filtering to remove frequencies above 1 or 2 kHz, and below 100 Hz or so. High frequencies do not have much voicing information and have significant noise energy, whereas low frequencies can have 50/60 Hz interference from power lines or non-linearities from some A/D subsystems that can also mislead a pitch estimation algorithm.

In addition to the noise in the very low frequencies and aspiration at high bands, the stationarity assumption is not so valid at high frequencies. Even a slowly changing pitch, say, nominal 100 Hz increasing 5 Hz in 10 ms, results in a fast-changing harmonic: the 30th harmonic at 3000 Hz changes 150 Hz in 10 ms. The corresponding short-time spectrum no longer shows peaks at those frequencies.

Because of this, it is advantageous to filter out such frequencies prior to the computation of the autocorrelation or normalized cross-correlation. If an FFT is used to compute the autocorrelation, this filter is easily done by setting to 0 the undesired frequency bins.

6.7.4. Pitch Tracking

Pitch tracking using the above methods typically fails in several cases:

- *Sub-harmonic errors*. If a signal is periodic with period T, it is also periodic with period $2T$, $3T$, etc. Thus, we expect the scores to be also high for the multiples of T, which can mislead the algorithm. Because the signal is never perfectly stationary, those multiples, or sub-harmonics, tend to have slightly lower scores than the fundamental. If the pitch is identified as $2T$, pitch halving is said to occur.
- *Harmonic errors*. If harmonic M dominates the signal's total energy, the score at pitch period T/M will be large. This can happen if the harmonic falls in a formant frequency that boosts its amplitude considerably compared to that of the other harmonics. If the pitch is identified as $T/2$, pitch doubling is said to occur.
- *Noisy conditions*. When the SNR is low, pitch estimates are quite unreliable for most methods.
- *Vocal fry*. While pitch is generally continuous, for some speakers it can suddenly change and even halve, particularly at the end of an unstressed voiced region. The pitch here is really not well defined and imposing smoothness constraints can hurt the system.
- *F0 jumps* up or down by an octave occasionally.
- *Breathy voiced speech* is difficult to distinguish from periodic background noise.
- *Narrow-band filtering* of unvoiced excitations by certain vocal tract configurations can lead to signals that appear periodic.

For these reasons, pitch trackers do not determine the pitch value at frame *m* based exclusively on the signal at that frame. For a frame where there are several pitch candidates with similar scores, the fact that pitch does not change abruptly with time is beneficial in disambiguation, because possibly the following frame has a clearer pitch candidate, which can help.

To integrate the normalized cross-correlation into a probabilistic framework, you can combine tracking with the use of a priori information [10]. Let's define $\mathbf{X} = \{\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_{M-1}\}$ as a sequence of input vectors for *M* consecutive frames centered at equally spaced time instants, say 10 ms. Furthermore, if we assume that the $\mathbf{x}_i$ are independent of each other, the joint distribution takes on the form:

$$f(\mathbf{X} \mid \mathbf{T}) = \prod_{i=0}^{M-1} f(\mathbf{x}_i \mid T_i) \tag{6.172}$$

where $\mathbf{T} = \{T_0, T_1, \ldots, T_{M-1}\}$ is the pitch track for the input. The *maximum a posteriori* (MAP) estimate of the pitch track is:

$$\mathbf{T}_{MAP} = \max_{\mathbf{T}} f(\mathbf{T} \mid \mathbf{X}) = \max_{\mathbf{T}} \frac{f(\mathbf{T}) f(\mathbf{X} \mid \mathbf{T})}{f(\mathbf{X})} = \max_{\mathbf{T}} f(\mathbf{T}) f(\mathbf{X} \mid \mathbf{T}) \tag{6.173}$$

according to Bayes' rule, with the term $f(\mathbf{X} \mid \mathbf{T})$ being given by Eq. (6.172) and $f(\mathbf{x}_i \mid T_i)$ by Eq. (6.169), for example.

The function $f(\mathbf{T})$ constitutes the *a priori* statistics for the pitch and can help disambiguate the pitch, by avoiding pitch doubling or halving given knowledge of the speaker's average pitch, and by avoiding rapid transitions given a model of how pitch changes over time. One possible approximation is given by assuming that the a priori probability of the pitch period at frame *i* depends only on the pitch period for the previous frame:

$$f(\mathbf{T}) = f(T_0, T_1, \ldots, T_{M-1}) = f(T_{M-1} \mid T_{M-2}) f(T_{M-2} \mid T_{M-3}) \cdots f(T_1 \mid T_0) f(T_0) \tag{6.174}$$

One possible choice for $f(T_t \mid T_{t-1})$ is to decompose it into a component that depends on T_t and another that depends on the difference $(T_t - T_{t-1})$. If we approximate both as Gaussian densities, we obtain

$$\ln f(T_t \mid T_{t-1}) = K' - \frac{(T_t - \mu)^2}{2\beta^2} - \frac{(T_t - T_{t-1} - \delta)^2}{2\gamma^2} \tag{6.175}$$

so that when Eqs. (6.170) and (6.175) are combined, the log-probability of transitioning to T_i at time *t* from pitch T_j at time *t* - 1 is given by

$$S_t(T_i, T_j) = \frac{(\max(0, \alpha_t(T_i)))^2}{2\sigma^2} - \frac{(T_i - \mu)^2}{2\beta^2} - \frac{(T_i - T_j - \delta)^2}{2\gamma^2} \tag{6.176}$$

so that the log-likelihood in Eq. (6.173) can be expressed as

$$\ln f(\mathbf{T}) f(\mathbf{X} \mid \mathbf{T}) = \left(\max(0, \alpha_0(T_0))\right)^2 + \max_{i_t} \sum_{t=1}^{M-1} S_t(T_{i_t}, T_{i_{t-1}}) \tag{6.177}$$

which can be maximized through dynamic programming. For a region where pitch is not supposed to change, $\delta = 0$, the term $(T_i - T_j)^2$ in Eq. (6.176) acts as a penalty that keeps the pitch track from jumping around. A mixture of Gaussians can be used instead to model different rates of pitch change, as in the case of Mandarin Chinese with four tones characterized by different slopes. The term $\left(T_i - \mu\right)^2$ attempts to get the pitch close to its expected value to avoid pitch doubling or halving, with the average μ being different for male and female speakers. Pruning can be done during the search without loss of accuracy (see Chapter 12).

Pitch trackers also have to determine whether a region of speech is voiced or unvoiced. A good approach is to build a statistical classifier with techniques described in Chapter 8 based on energy and the normalized cross-correlation described above. Such classifiers, *i.e.*, an HMM, penalize jumps between voiced and unvoiced frames to avoid voiced regions having isolated unvoiced frame inside and vice versa. A threshold can be used on the a posteriori probability to distinguish voiced from unvoiced frames.

6.8. Historical Perspective And Future Reading

In 1978, Lawrence R. Rabiner and Ronald W. Schafer [38] wrote a book summarizing the work to date on digital processing of speech, which remains a good source for the reader interested in further reading in the field. The book by Deller, Hansen and Proakis [9] includes more recent work and is also an excellent reference. O’Shaughnessy [33] also has a thorough description of the subject. Malvar [25] covers filterbanks and lapped transforms extensively.

The extensive wartime interest in sound spectrography led Koenig and his colleagues at Bell Laboratories [22] in 1946 to the invaluable development of a tool that has been used for speech analysis since then: the spectrogram. Potter et al. [35] showed the usefulness of the analog spectrogram in analyzing speech. The spectrogram facilitated research in the field and led Peterson and Barney [34] to publish in 1952 a detailed study of formant values of different vowels. The development of computers and the FFT led Oppenheim, in 1970 [30], to develop digital spectrograms, which imitated the analog counterparts.

The MIT Acoustics Lab started work in speech in 1948 with Leo R. Beranek, who in 1954 published the seminal book *Acoustics*, where he studied sound propagation in tubes. In 1950, Kenneth N. Stevens joined the lab and started work on speech perception. Gunnar Fant visited the lab at that time and as a result started a strong speech production effort at KTH in Sweden.

The 1960s marked the birth of digital speech processing. Two books, Gunnar Fant’s *Acoustical Theory of Speech Production* [13] in 1960 and James Flanagan’s *Speech Analysis: Synthesis and Perception* [14] in 1965, had a great impact and sparked interest in the field. The advent of the digital computer prompted Kelly and Gertsman to create in 1961 the first digital speech synthesizer [21]. Short-time Fourier analysis, cepstrum, LPC analysis, pitch and formant tracking, and digital filterbanks were the fruit of that decade.

Short-time frequency analysis was first proposed for analog signals by Fano [11] in 1950 and later by Schroeder and Atal [42].

The mathematical foundation behind linear predictive coding dates to the auto-regressive models of George Udny Yule (1927) and Gilbert Walker (1931), which led to the well-known Yule-Walker equations. These equations resulted in a Toeplitz matrix, named after Otto Toeplitz (1881 - 1940) who studied it extensively. N. Levinson suggested in 1947 an efficient algorithm to invert such a matrix, which J. Durbin refined in 1960 and is now known as the Levinson-Durbin recursion. The well-known LPC analysis consisted of the application of the above results to speech signals, as developed by Bishnu Atal [4], J. Burg [7], Fumitada Itakura and S. Saito [19] in 1968, Markel [27] and John Makhoul [24] in 1973.

The cepstrum was first proposed in 1964 by Bogert, Healy and John Tukey [6] and further studied by Alan V. Oppenheim [29] in 1965. The popular mel-frequency cepstrum was proposed by Davis and Mermelstein [8] in 1980, combining the advantages of cepstrum with knowledge of the non-linear perception of frequency by the human auditory system that had been studied by E. Zwicker [47] in 1961.

The study of digital filterbanks was first proposed by Schafer and Rabiner in 1971 for IIR and in 1975 for FIR filters.

Formant tracking was first investigated by Ken Stevens and James Flanagan in the late 1950s, with the foundations for most modern techniques being developed by Schafer and Rabiner [40], Itakura [20], and Markel [26]. Pitch tracking through digital processing was first studied by B. Gold [15] in 1962 and then improved by A. M. Noll [28], M. Schroeder [41], and M. Sondhi [44] in the late 1960s.

REFERENCES

[1] Acero, A., "Formant Analysis and Synthesis using Hidden Markov Models," *Eurospeech*, 1999, Budapest pp. 1047-1050.

[2] Atal, B.S., *Automatic Speaker Recognition Based on Pitch Contours*, PhD Thesis 1968, Polytechnic Institute of Brooklyn, .

[3] Atal, B.S. and L. Hanauer, "Speech Analysis and Synthesis by Linear Prediction of the Speech Wave," *Journal of the Acoustical Society of America*, 1971, **50**, pp. 637-655.

[4] Atal, B.S. and M.R. Schroeder, "Predictive Coding of Speech Signals," *Report of the 6th Int. Congress on Acoustics*, 1968, Tokyo, Japan.

[5] Berouti, M.G., D.G. Childers, and A. Paige, "Glottal Area versus Glottal Volume Velocity," *Int. Conf. on Acoustics, Speech and Signal Processing*, 1977, Hartford, Conn pp. 33-36.

[6] Bogert, B., M. Healy, and J. Tukey, "The Quefrency Alanysis of Time Series for Echoes," *Proc. Symp. on Time Series Analysis*, 1963, New York, J. Wiley pp. 209-243.

[7] Burg, J., "Maximum Entropy Spectral Analysis," *Proc. of the 37th Meeting of the Society of Exploration Geophysicists*, 1967.

[8] Davis, S. and P. Mermelstein, "Comparison of Parametric Representations for Monosyllable Word Recognition in Continuously Spoken Sentences," *IEEE Trans. on Acoustics, Speech and Signal Processing*, 1980, **28**(4), pp. 357-366.

[9] Deller, J.R., J.H.L. Hansen, and J.G. Proakis, *Discrete-Time Processing of Speech Signals*, 2000, IEEE Press.

[10] Droppo, J. and A. Acero, "Maximum a Posteriori Pitch Tracking," *Int. Conf. on Spoken Language Processing*, 1998, Sydney, Australia pp. 943-946.

[11] Fano, R.M., "Short-time Autocorrelation Functions and Power Spectra," *Journal of the Acoustical Society of America*, 1950, **22**(Sep), pp. 546-550.

[12] Fant, G., "On the Predictability of Formant Levels and Spectrum Envelopes from Formant Frequencies" in *For Roman Jakobson*, M. Halle, Editor 1956, The Hague, NL, pp. 109-120, Mouton & Co.

[13] Fant, G., *Acoustic Theory of Speech Production*, 1970, The Hague, NL, Mouton.

[14] Flanagan, J., *Speech Analysis Synthesis and Perception*, 1972, New York, Springer-Verlag.

[15] Gold, B., "Computer Program for Pitch Extraction," *Journal of the Acoustical Society of America*, 1962, **34**(7), pp. 916-921.

[16] Hermansky, H., "Perceptual Linear Predictive (PLP) Analysis of Speech," *Journal of the Acoustical Society of America*, 1990, **87**(4), pp. 1738-1752.

[17] Hess, W., *Pitch Determination of Speech Signals*, 1983, New York, Springer-Verlag.

[18] Itakura, F., "Line Spectrum Representation of Linear Predictive Coefficients," *Journal of the Acoustical Society of America*, 1975, **57**(4), pp. 535.

[19] Itakura, F. and S. Saito, "Analysis Synthesis Telephony Based on the Maximum Likelihood Method," *Proc. 6th Int. Congress on Acoustics*, 1968, Tokyo, Japan.

[20] Itakura, F. and S. Saito, "A Statistical Method for Estimation of Speech Spectral Density and Formant Frequencies," *Elec. and Comm. in Japan*, 1970, **53-A**(1), pp. 36-43.

[21] Kelly, J.L. and L.J. Gerstman, "An Artificial Talker Driven From Phonetic Input," *Journal of Acoustical Society of America*, 1961, **33**, pp. 835.

[22] Koenig, R., H.K. Dunn, and L.Y. Lacy, "The Sound Spectrograph," *Journal of the Acoustical Society of America*, 1946, **18**, pp. 19-49.

[23] Krishnamurthy, A.K. and D.G. Childers, "Two Channel Speech Analysis," *IEEE Trans. on Acoustics, Speech and Signal Processing*, 1986, **34**, pp. 730-743.

[24] Makhoul, J., "Spectral Analysis of Speech by Linear Prediction," *IEEE Trans. on Acoustics, Speech and Signal Processing*, 1973, **21**(3), pp. 140-148.

[25] Malvar, H., *Signal Processing with Lapped Transforms*, 1992, Artech House.

[26] Markel, J.D., "Digital Inverse Filtering - A New Tool for Formant Trajectory Estimation," *IEEE Trans. on Audio and Electroacoustics*, 1972, **AU-20**(June), pp. 129-137.

[27] Markel, J.D. and A.H. Gray, "On Autocorrelation Equations as Applied to Speech Analysis," *IEEE Trans. on Audio and Electroacoustics*, 1973, **AU-21**(April), pp. 69-79.

[28] Noll, A.M., "Cepstrum Pitch Determination," *Journal of the Acoustical Society of America*, 1967, **41**, pp. 293--309.

[29] Oppenheim, A.V., *Superposition in a Class of Nonlinear Systems*, 1965, Research Lab. Of Electronics, MIT, Cambridge, Massachusetts.

[30] Oppenheim, A.V., "Speech Spectrograms Using the Fast Fourier Transform," *IEEE Spectrum*, 1970, **7**(Aug), pp. 57-62.

[31] Oppenheim, A.V. and D.H. Johnson, "Discrete Representation of Signals," *The Proc. of the IEEE*, 1972, **60**(June), pp. 681-691.

[32] Oppenheim, A.V., R.W. Schafer, and T.G. Stockham, "Nonlinear Filtering of Multiplied and Convolved Signals," *Proc. of the IEEE*, 1968, **56**, pp. 1264-1291.

[33] O'Shaughnessy, D., *Speech Communication -- Human and Machine*, 1987, Addison-Wesley.

[34] Peterson, G.E. and H.L. Barney, "Control Methods Used in a Study of the Vowels," *Journal of the Acoustical Society of America*, 1952, **24**(2), pp. 175-184.

[35] Potter, R.K., G.A. Kopp, and H.C. Green, *Visible Speech*, 1947, New York, D. Van Nostrand Co. Republished by Dover Publications, Inc. 1966.

[36] Press, W.H., *et al.*, *Numerical Recipes in C*, 1988, New York, NY, Cambridge University Press.

[37] Rabiner, L.R., "On the Use of Autocorrelation Analysis for Pitch Detection," *IEEE Trans. on Acoustics, Speech and Signal Processing*, 1977, **25**, pp. 24-33.

[38] Rabiner, L.R. and R.W. Schafer, *Digital Processing of Speech Signals*, 1978, Englewood Cliffs, NJ, Prentice-Hall.

[39] Rosenberg, A.E., "Effect of Glottal Pulse Shape on the Quality of Natural Vowels," *Journal of the Acoustical Society of America*, 1971, **49**, pp. 583-590.

[40] Schafer, R.W. and L.R. Rabiner, "System for Automatic Formant Analysis of Voiced Speech," *Journal of the Acoustical Society of America*, 1970, **47**, pp. 634--678.

[41] Schroeder, M., "Period Histogram and Product Spectrum: New Methods for Fundamental Frequency Measurement," *Journal of the Acoustical Society of America,* 1968, **43**(4), pp. 829-834.

[42] Schroeder, M.R. and B.S. Atal, "Generalized Short-Time Power Spectra and Autocorrelation," *Journal of the Acoustical Society of America*, 1962, **34**(Nov), pp. 1679-1683.

[43] Shikano, K., K.-F. Lee, and R. Reddy, "Speaker Adaptation through Vector Quantization," *IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, 1986, Tokyo, Japan pp. 2643-2646.

[44] Sondhi, M.M., "New Methods for Pitch Extraction," *IEEE Trans. on Audio and Electroacoustics*, 1968, **16**(June), pp. 262-268.

[45] Talkin, D., "A Robust Algorithm for Pitch Tracking" in *Speech Coding and Synthesis*, W.B. Kleijn and K.K. Paliwal, eds. 1995, Amsterdam, pp. 485-518, Elsevier.

[46] Yegnanarayana, B. and R.N.J. Veldhuis, "Extraction of Vocal-Tract System Characteristics from Speech Signals," *IEEE Trans. on Speech and Audio Processing*, 1998, **6**(July), pp. 313-327.

[47] Zwicker, E., "Subdivision of the Audible Frequency Range into Critical Bands," *Journal of the Acoustical Society of America*, 1961, **33**(Feb), pp. 248.